OPTIMUM COMPONENT DESIGN IN N-STAGE SERIES SYSTEMS TO MAXIMIZE THE RELIABILITY UNDER BUDGET CONSTRAINT

THESIS

Huseyin Gozebe, First Lieutenant, TUAF

AFIT/GOR/ENS/03-08

DEPARTMENT OF THE AIR FORCE
AIR UNIVERSITY
AIR FORCE INSTITUTE OF TECHNOLOGY

Wright-Patterson Air Force Base, Ohio

AFIT/GOR/ENS/03-08

OPTIMUM COMPONENT DESIGN IN N-STAGE SERIES SYSTEMS TO MAXIMIZE THE RELIABILITY UNDER BUDGET CONSTRAINT

THESIS

Presented to the Faculty

Department of Operational Sciences

Graduate School of Engineering and Management

Air Force Institute of Technology

Air University

Air Education and Training Command

In Partial Fulfillment of the Requirements for the

Degree of Master of Science in Operations Research

Huseyin Gozebe, BS

First Lieutenant, TUAF

March 2003

AFIT/GOR/ENS/03-08

OPTIMUM COMPONENT DESIGN IN N-STAGE SERIES SYSTEMS TO MAXIMIZE THE RELIABILITY UNDER BUDGET CONSTRAINT

Huseyin Gozebe, BS
First Lieutenant, TUAF

Approved:

Stephen P. Chambal (Advisor)	date
Robin Nicole Benton (Reader)	date

Acknowledgments

I would like to thank my advisor Capt. Stephen Chambal and reader Maj. Nicole Benton for their guidance and support. Also, I would like to thank my wife for her endless love and patience during my time in AFIT. In addition, I would like to thank my mother for her commitment and love throughout my life. I also appreciate the support from all of my friends, especially Vedat Akgun and Vic Grazier.

Huseyin Gozebe

Table of Contents

Page

Page

Page

List of Figures

Page

Page

List of Tables

Page

Abstract

This research is intended to find a new heuristic for solving the reliability optimization of n-stage series system. The new heuristic will make the initial system design and use redundancy to improve system reliabilility level. The limited capacity of common optimization tools requires the new heuristic to be coded in VBA programming language. Moreover, the known reliability optimization heuristic, marginal analysis will be coded so that the difference between two heuristics can be seen and design engineers can use the best result in their applications.

OPTIMUM COMPONENT DESIGN IN N-STAGE SERIES SYSTEMS TO MAXIMIZE THE RELIABILITY UNDER BUDGET CONSTRAINT

I. Introduction

1.1 General Issue

Every system has its own reliability to achieve working goals. The reliability of the entire system depends on both the structure of the system and the reliability of every individual component. The structure is defined as series, parallel, series-parallel, parallel-series, mixed, or complex. The design engineer tries to construct subsystems and individual components in subsystems to obtain the maximum system reliability. Most of the time, however, this is not the only important thing. The engineer must consider the cost of the system under the given resource constraints.

Every subsystem or component will cause an increase in cost and reduction of limited resources. The system can be designed regardless of cost with redundant components to get the maximum reliability. Strictly looking at maximizing reliability is infeasible, as corporations, along with the Air Force, have limited budgets. The design engineer must take these constraints into consideration when developing a new system.

1.2 Background

Reliability allocation has an important role in systems design and is the focus of this research. The goal of reliability allocation is to maximize reliability while taking cost into consideration. "The objective of this allocation is to use the reliability model to

assign reliability to the subsystems so as to achieve a specified reliability goal for the system. Reliability and design engineers must translate overall system performance, including reliability, into component performance, including reliability. The process of assigning reliability requirements to individual components to attain the specified system reliability is called reliability allocation" (Kapur and Lamberson, 1977). With resource and cost constraints, the problem becomes a multi-objective optimization problem. The goal is to maximize reliability and minimize cost consumed. One way to view this problem is with only one objective function, maximizing system reliability, with money as the only constraint. Therefore, all money can be used instead of simply minimizing its consumption.

Kapur and Lamberson discuss the complexity of reliability allocation, along with its advantages in the following passage.

1. The reliability allocation program forces the designer to understand and develop the relationship between component, subsystem, and system reliabilities. This leads to an understanding of the basic reliability problems inherent in the design.

2. The designer is obliged to consider reliability equally with other system parameters such as weight, cost and performance characteristics.

3. The contractor is required to meet the reliability goals in military contracts. In this situation, the reliability allocation program results in improved design, manufacturing methods, and testing procedures.

1.3 Problem Statement

This research will cover optimal component allocation for system design to maximize the reliability of the system under certain cost constraints. At the beginning there are sets of components for each subsystem, and after selecting the appropriate number of components from these sets to construct the basic system, then redundancy

will be used to improve the overall system reliability. This means that the proper components will be added into subsystems in the basic structure. The system design will take place in stages, with the system reliability calculated at each point and compared to a newly designed or adjusted layout of components and subsystems. By examining the system structure and the overall reliability, the number of redundant components and the individual component reliabilities can be calculated.

1.4 Research Objectives

This study will examine the selection of the appropriate number of components for each subsystem from a set for each subsystem and then adding redundant components into the subsystems in parallel structure. Cost constraint will limit the number of redundant components. A heuristic approach will be developed to determine the final design for maximum reliability. The aim for this heuristic is to avoid formulation complexity of complicated systems while trying to obtain optimum reliability. The new heuristic will be a greedy construction heuristic, which improves the solution at the end of each iteration. Later, this heuristic will be compared to other solution techniques for optimal resource allocation.

Visual Basic for Applications (VBA) will be used for coding this new heuristic method to provide an easy to use tool for design engineers. VBA gives the advantage of thousands of integer variables to use, and it can be used even with k-out-of-n systems and is easy to find in every office where PCs and EXCEL are available.

1.5 Research Methodology

Known heuristic techniques will be integrated with resource allocation to obtain a new method for optimal system design with respect to maximizing reliability. For system evaluation, the system will be separated and solved at the subsystem level. Our desired cost and reliability analysis will be integrated as the subsystem evaluation is merged into an overall system solution. The results for the developed heuristic will be compared with known methods from past research by using hypothesis tests. The new heuristic will be coded using VBA and EXCEL.

1.6 Outline of Thesis

This thesis is divided into five Chapters: Introduction, Literature Review, Methodology, Results, Conclusions and Recommendations. Short descriptions of these are as follows:

Chapter 1: Introduction: In this chapter, the background, problem statement, research objectives, and research methodology are briefly discussed.

Chapter 2: Literature Review: In this chapter, the theoretical background of this thesis will be researched to provide an appropriately strong and useful foundation.

Chapter 3: Methodology: The new heuristic method will be constructed after explaining the exact problem, and the VBA coding will be written for this new method.

Chapter 4: Results: The results for the new heuristic will be demonstrated on a small example problem. The heuristic will be compared to other techniques by using hypothesis tests, and a comparison will be completed.

Chapter 5: Conclusions and Recommendations: In this chapter, a brief overview will be written highlighting the most important aspects of the thesis and suggestions for future work will be recommended.

II. Literature Review

2.1 Introduction

To work**,** all systems must be in a certain design structure depending on their special function. Subsystems, or stages and even individual components, have connection links to each other in their own systems. Design engineers construct systems by taking special principles into consideration. The reliability level of every subsystem, or stage and component, affects system designs, and the engineer designs the system to obtain maximum reliability. This chapter includes a review of heuristics; engineering design, design for reliability, reliability block diagrams, reliability optimization tasks, reliability improvement methods, redundancy, reliability allocation methods, visual basic for application (VBA), and finally, a summary section.

2.2 Heuristics

Reeves, 1995, gives the definition of heuristic as "A heuristic is a technique which seeks good (i.e. near optimal) solutions at a reasonable computational cost without being able to guarantee either feasibility or optimality, or even in many cases to state how close to optimality a particular feasible solution is." Muller and Merbach discuss the heuristics as "In OR, the term 'heuristic' is usually understood in the sense of an iterative algorithm which does not converge toward the ('optimal' or even 'only a feasible') solution of a problem". Therefore, heuristics can be said to be algorithms without proven convergence to the optimum solution. One of the reasons for using heuristics is the

computation complexity of problems. Heuristics can reasonably reduce the amount of computation for computers so that the near optimal solution can be obtained quickly. Heuristics are faster and need less storage in computer memory than optimization tools require. Another important feature of heuristics is that they are required if there is a lack of computer code for a specific problem. This research will handle this issue later.

Along with an understanding of heuristics, this problem also requires an understanding of engineering design. The following section discusses engineering design so that the importance of reliability to design can be understood effectively.

2.3 Engineering Design

Above everything else, engineering design has to be explained because optimum redundant component allocation requires that knowledge. For engineering design, there are various definitions in literature. For the purpose of this research the following definition will be accepted, "Engineering design is the activity in which various methods and scientific principles are used to decide the selection of materials and the placement of these materials to develop an item that fulfills specified requirements." (Dhillion, 1985). What is the role of reliability in engineering design? The answer to this question is that "reliability is considered to be one of the most important quality characteristics of technical products. Reliability assurance should therefore be one of the most important topics during the engineering design process."(Hoyland, Rausand, 1994).

The goal of engineering design is customer satisfaction in the product, and to achieve this ultimate goal, the design engineer must take many factors into consideration. "The three key factors in achieving user satisfaction in most designs are: 1) reliability, the

ability of the product to continually perform its intended functions; 2) maintainability and testability and supportability if and when the product fails, the ease with which the item can be placed back in service; and 3) achieving factors 1 and 2 at a minimum cost. Satisfying these three factors will assure user satisfaction."(Jones, 1988). Clearly, reliability has a special concern in the design process and requires an extensive knowledge to construct systems with the desired reliability levels.

2.4 Design for Reliability

"To a large degree, reliability is an inherent attribute of a system, component, or product. As such, it is an important consideration in the engineering design process. When the life-cycle costs of a system are being analyzed, reliability plays an important role as a major driver of these costs and has considerable influence on system performance."(Ebeling, 1997). Therefore, maximum reliability under cost constraints is very important for the life-cycle cost of the system. To achieve this goal, we have to take the final reliability design of the system into consideration. Reliability design is further defined since it is very important for a designer.

"Reliability design is an iterative process that begins with the specification of reliability goals consistent with cost and performance objectives. This requires consideration of the life-cycle costs of the system and the effect that reliability has on overall cost and system effectiveness." (Ebeling, 1997). A reliability design process algorithm should consider the overall system effectively. With this algorithm we have to analyze the all-important points for system reliability. Ebeling gives a good algorithm:

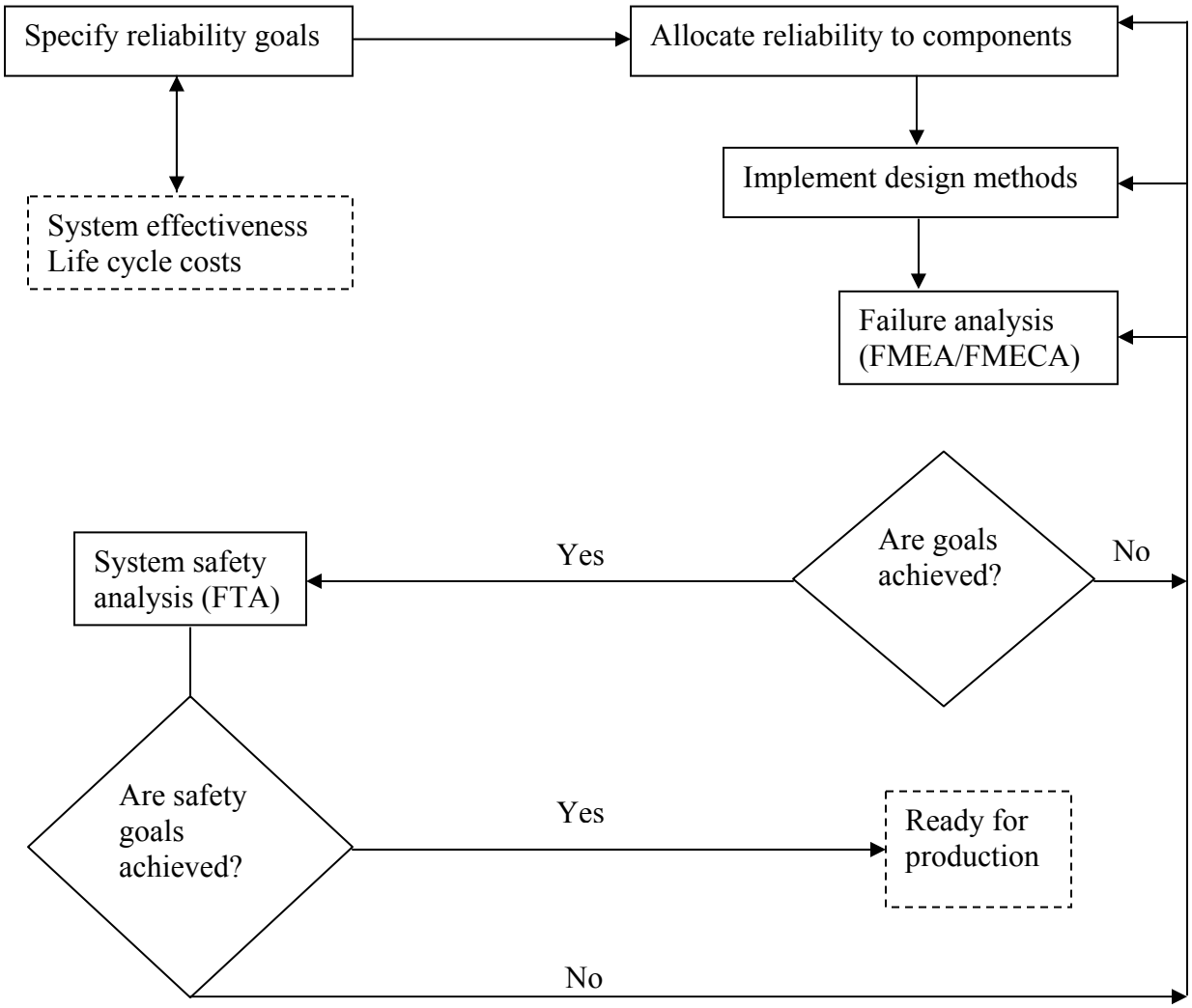


Figure 1 The Reliability Design Process (Ebeling, 1997)

Clearly, allocation of reliability to components is very important, and it occurs at an early stage of the design process as well. This means, reliability of the system and component reliability levels have to be defined before everything else in the design process. This step will have special and painstaking effort in itself to create the component design puzzle. The order of components to obtain subsystems and the order

of the subsystems have to be arranged to obtain the desired system, which has maximum reliability under the given cost constraint.

This can be difficult, and involves the implementation of a reliability block analysis. First of all, we have to determine individual component and part requirements, and then we can use several design methods to achieve our desired and final goal. "These methods include the proper selection of parts and material, stress-strength analysis, derating, simplification, identification of technologies, and use of redundancy." (Ebeling, 1997). This study will search for the proper selection of parts and material with a certain reliability level and redundancy use.

While implementing the initial design step, one must take all methods and different block designs into consideration as much as possible to get a detailed final design. After reaching the final desired model, there is no easy way to break up the model and add new redundant components or use other techniques such as simplification or new technology.

Another important point is how the design engineer behaves under some certain budget constraint when constructing a system design. The major question is, "Which components with which reliability level and specific cost have to be put in system?" The answer to this question requires a deep search for all possible combinations of components under each subsystem. The design engineer has to search for the optimum design to get maximum reliability while still operating under the budget constraint.

What all possible combinations mean is that subsystems and components can be in series, parallel or other structures. To understand this more deeply, reliability block diagrams have to be explained.

2.5 Reliability Block Diagrams

"The first step in evaluating a system's reliability is to construct a reliability block diagram, which is a graphical representation of the components of the system and how they are connected. (Elsayed, 1996). From the aspect of success, literature has another definition of the reliability block diagram. **"**A reliability block diagram is a success-oriented network describing the function of the system. If the system has more than one function, each function is considered individually, and a separate reliability block diagram is established for each system function."(Hoyland, Rausand, 1994).

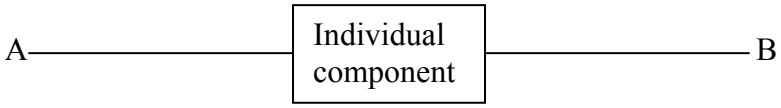


Figure 2 An Individual Component

In figure 2, a block illustrates an individual component. To be a reliable component, there should be connection between points A and B, and in this case it can be technically said that this individual component is functioning. For more complex systems, which include many components connected in various structures, at least some of them have to be in a functioning position so that the connection occurs between the starting point and the end point of the entire system. Moreover, it can be written that the state of the individual component can be represented with a binary random variable X_i, where

$$X_i = \begin{cases} 1, \text{ the individual component is functioning.} \\ 0, \text{ the individual component is not functioning.} \end{cases}$$

After describing individual component reliability, overall system reliability can be examined. Several different approaches for evaluating the overall system are found in reliability literature, but only four of them are the concern of this research. The new heuristic will be designed to solve the systems in series-parallel structure; therefore, series systems, parallel systems, series-parallel systems, and k-out-of-n systems will be discussed in the following sections.

2.5.1 Series Systems.

Series systems are constructed with n components (or subsystems), which are connected in a series structure. If one of the components fails, the entire system fails immediately. The following notations and equations come from Elsayed, 1996:

X_i = The i^{th} unit is operational.

$\bar{X}_i$ = Failure of the i^{th} unit.

$P(X_i)$ = Probability that i^{th} unit is operational.

$P(\bar{X}_i)$ = Probability that i^{th} is not operational (failed).

R = Reliability of the system, and

P_f = Unreliability of the system ($Pf = 1 - R$).

If all components work, which is the intersection of reliability of working components, the overall system works; this is expressed in the following equation:

$$R = P(X_1 X_2 X_3 \ldots) \quad \textbf{(1)}$$

If all of the components are independent, equation 1 can be rewritten as the following equation:

$$R = P(X_1)P(X_2)...P(X_n) \text{ or,}$$

$$R = \prod_{i=1}^{n} P(X_i) \qquad \textbf{(2)}$$

The reliability of a series system is always less than or equal to the lowest reliability of an individual component in the overall system.

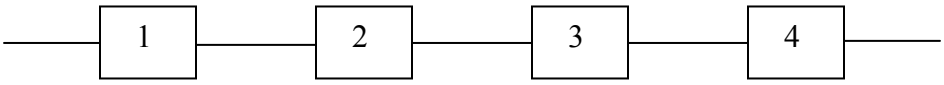


Figure 3 Series System while n = 4

Clearly, all subsystems have to be in working condition for the overall system to work, but in some other reliability block diagrams, this isn't required for the overall system to work. One of these diagrams is a parallel system.

2.5.2 Parallel Systems.

In parallel systems, all components are in parallel structure, and if one of these components works, the whole system will work. This is the union of reliability of working components, and it is expressed as follows:

$$R = P(X_1 \cup X_2 \cup ...X_n) \qquad \textbf{(3)}$$

Alternatively,

$R = 1 - P_f$ if all components are independent,

$$R = 1 - \prod_{i=1}^{n} P(\bar{X}_i) \qquad \textbf{(4)}$$

If the components are identical, then:

$$R = 1 - P(1-p)^n \quad \textbf{(5)}$$

where p is the probability that a component is working.

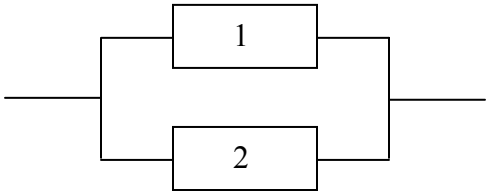


Figure 4 Parallel System while n = 2

The reliability of a parallel system is greater than the reliability of the component, which has the highest reliability in the system.

Series and parallel systems are basic elements for more complicated systems, which are parallel-series, series-parallel, and mixed-parallel systems. These complicated systems use the same reliability computing formulations, but at the same time they are reduced to a simplified version. The following sections will explain these systems.

2.5.3 Series-Parallel Systems.

Series-parallel systems have n subsystems in series, and these series systems have m units in parallel in each subsystem as shown in figure 5:

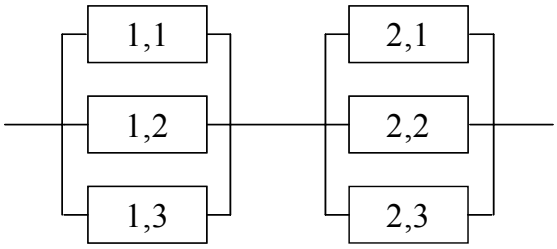


Figure 5 Series-Parallel System while m = 3 and n = 2

$$R = \prod_{i=1}^{n}[1 - \prod_{j=1}^{m}(1 - P(X_{ij}))] \quad \textbf{(6)}$$

When all units are identical and the reliability of a single unit is p, then

$$R = [1 - (1 - p)^m]^n \quad \textbf{(7)}$$

2.5.4 K-Out-of-n Systems.

Ebeling, 1997 discusses this subject with the following words.

> A generalization of n parallel components occurs when a requirement exists for k out of n identical and independent components to function for the system to function. Obviously k ≤ n. If k = 1, complete redundancy occurs, and if k = n, the n components are, in effect, in series. The reliability can be obtained from the binomial probability distribution.
>
> If each component is viewed as an independent trial with R (its reliability) as a constant probability of success, then

$$P(x) = (n, x) * (R^x) * (1 - R)^{(n-x)} \quad \textbf{(8)}$$

> is the probability of exactly x components operating. This is true since

$$(n, x) = \frac{n!}{x * (n - x)!} \quad \textbf{(9)}$$

> is the number of ways (arrangements) in which x successes (non-failures) can be obtained from n components. $(R^x)(1 - R)^{n-x}$ is the probability of x successes and n-x failures for a single arrangement of successes and failures. Therefore

$$R_s = \sum_{x=k}^{n} P(x) \quad \textbf{(10)}$$

> is the probability of k or more successes from among the n components.

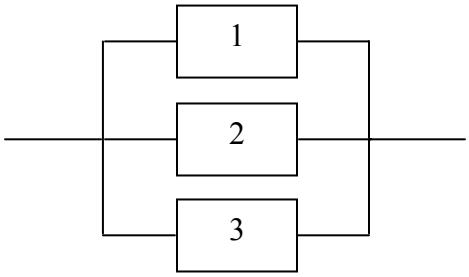


Figure 6 A 2-out-of-3 System

Figure 6 is an illustration of a 2-out-3 systems as defined by graphical representation of the block diagram if any of the two components work, then the entire system can work. Meanwhile, this system can also be illustrated as follows when describing the underlying functioning of the system.

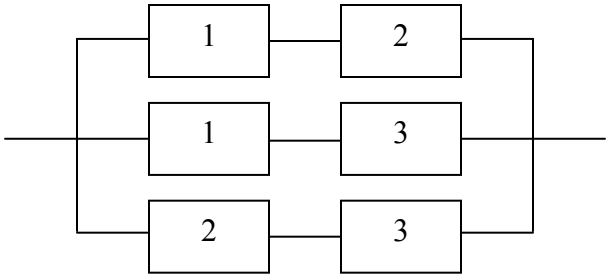


Figure 7 Symbolic Illustration of K-out-of-N System

If at least one path in the figure works, then the entire system works. Figure 6 and 7 are the same k-out-of-n system simply displayed in two different ways. Figure 7 specifically illustrates how the system is functioning.

Up to this point, mainly, engineering design, the importance of reliability in the engineering design, and reliability block diagrams are discussed. Along with an

understanding of previous sections, the optimization part of this research will be discussed in the following sections.

2.6 Reliability Optimization Tasks

In literature, there are two optimization tasks, the first one is minimizing cost while keeping a minimum reliability level, and the second one is maximizing reliability under given budget constraints. Ushakov discusses this issue with following notation, and words:

$C(X_1, X_2 ... X_n)$ = Total system cost when its subsystems have $X_1, X_2 ... X_n$ redundant units.

C_0 = A given acceptable level of a single resource expenditure for the whole system.

$R(X_1, X_2 ... X_n)$ = System reliability index of type R when the numbers of redundant units in its subsystems are $X_1, X_2 ... X_n$.

R_0 = Objective function value of the reliability index R.

$$Q(X_1, X_2 ... X_n) = 1 - R(X_1, X_2 ... X_n)$$

$$Q_0 = 1 - R_0$$

$$L(X_1, X_2 ... X_n) = -\log R(X_1, X_2 ... X_n)$$

$$L_0 = -\log R_0 .$$

1) One type, minimum cost, in determination of the least expensive suite of redundant units that achieves an objective system reliability index, R. This "direct" type may be written

$$Min\{C(X_1, X_2 ... X_n) / R(X_1, X_2 ... X_n) \geq R_0\},$$

where the inequality may be replaced by $Q(X_1, X_2 ... X_n) < Q_0$

or $L(X_1, X_2 ... X_n) < L_0$.

2) The other type, maximum effectiveness, is determination of the suite of redundant units that produces the highest system reliability index that can be achieved for a total system cost that is not larger than a given constraint cost, C_0. The "inverse" type may be written

$$Max\{R(X_1, X_2 ... X_n) / C(X_1, X_2 ... X_n) \le C_0\},$$

where $R(X_1, X_2 ... X_n)$ might be replaced with $Q(X_1, X_2 ... X_n)$ or $L(X_1, X_2 ... X_n)$ when the operator max is changed to min.

These two tasks are desired goals of design engineers. The following sections will discuss how design engineers can reach these two goals with special methods.

2.7 Reliability Improvement Methods

There are mainly two reliability improvement methods in literature. "The reliability of a multistage series system can be improved by: 1) using more reliable components, or 2) adding redundant components in parallel. A designer is required to minimize system cost and the system weight, while simultaneously maximizing the system reliability."(Dhingra, 1992)

Redundancy is the main concern of this research, so it should be explained in detail in the next section. Meanwhile, redundancy will be used to improve the system reliability after selecting the appropriate number of components for each subsystem from a set of components of each subsystem as basic structure.

2.7.1 Redundancy.

"Redundancy is defined as the use of additional components or units beyond the number actually required for the satisfactory operation of a system for the purpose of improving its reliability. A series system has no redundancy since a failure of any

component causes failure of the entire system, whereas a parallel system has redundancy since the failure of a component (or possibly more) does not result in a system failure. Similarly, consecutive-k-out-of-n: F, k-out-n: F, parallel-series, and series-parallel systems have redundancy."(Elsayed, 1996). The components in this research are assumed to be in two states: in a working or failure state any time while the system works. Because of the two-state feature of components, to simply increase the component number in a parallel structure by the redundancy method will increase the reliability of the overall system.

"Redundancy may play an important role in the design process, especially when individual component reliabilities have already been established through an existing design or as a result of various uncontrollable failure modes. When it is impossible to achieve the desired component reliability through inherent component design, redundancy may provide the only alternative."(Ebeling, 1997). Therefore, redundancy has special concern in system design.

There are two kinds of redundancy in literature, "active and inactive redundancy. In active redundancy, all redundant components are in operation and are sharing the load with the main unit. Under nonactive standby, the redundant components do not share any amount of the load with the main components, and they start operating only when one or more operating components fail."(Elsayed, 1996). This research will be concerned only with the active redundancy situation.

In literature, there are many techniques that use redundancy to obtain optimum system reliability under budget constraints. With these techniques, the design engineer computes the best system design to maximize reliability without exceeding the budget

constraint. Using reliability allocation methods does this, and the next sections will discuss the reliability allocation methods within the scope of this research. Since this research will examine improving the reliability level of series-parallel systems, the reliability allocation methods related with series-parallel systems will be examined.

2.8 Reliability Allocation Methods

There are several reliability allocation methods in the literature. In this research, the methods of parallel systems, series-parallel systems, steepest descent, and marginal analysis will be examined. As a general approach Ebeling discusses the problem with the following formulation and words:

> Ideally, reliability allocation should be accomplished in a least-cost manner. If each component has a current reliability R_i where $\prod R_i < R^*$, we may be interested in solving the following problem:
>
> $$\text{Min z} = \sum_{i=1}^{n} C_i(x_i) \qquad \textbf{(11)}$$
>
> Subject to:
>
> $$\prod_{i=1}^{n} (R_i + x_i) \geq R^* \qquad \textbf{(12)}$$
>
> $$0 < R_i + x_i \leq B_i < 1 \quad \text{where i= 1…n} \qquad \textbf{(13)}$$
>
> where x_i is the increase in reliability of the ith component, $C_i(x_i)$ is the corresponding cost of achieving this growth, and B_i is an upper bound on the attainable component reliability.

This optimization can be solved for both linear and non-linear situations with known techniques. This formulation is used for the first optimization task; however, the

second optimization task can be formulated as told in reliability optimization tasks section. But, more complicated systems and optimization tools with limited capacity require heuristics and computer coding to solve the complicated systems, especially k-out-n systems are the concern of design engineers. Now more specific methods will be examined for our problem.

2.8.1 Parallel Systems Method.

Reliability block diagrams have been described. These will now be examined with the following aspect of our problem. Dhillion discusses this method with following formulation and words.

> We first ask, how much one should pay to increase reliability of a basic system from R_b to R_p when only parallel redundancy of the basic system can be used. Although, the basic system could have multiple elements in series form, in this text we consider only one element basic system.
>
> The independent and identical n-units parallel system reliability R_p is given by:

$$R_p = 1 - (1 - R_b)^n \quad \textbf{(14)}$$

where R_b is the unit or basic system reliability.
Equation 14 can be rewritten:

$$(1 - R_b)^n = 1 - R_p \quad \textbf{(15)}$$

Taking the logarithm of eq. 15 and rearranging

$$n = \frac{\log(1 - R_p)}{\log(1 - R_b)} \quad \textbf{(16)}$$

Now suppose that

$$C_s = nC_b \quad \textbf{(17)}$$

where C_s is the cost of the parallel or final (total) system, C_b is the unit cost of the original basic system. Substituting eq. 17 in equation 16 we get

$$n = \left(\frac{C_s}{C_b}\right) = \frac{\log(1 - R_p)}{\log(1 - R_b)} \qquad (18)$$

The above relationship can be used to obtain the cost figure necessary to increase R_b to the desired final system reliability, R_p.

This formulation in 18 is used for the same components. For instance, there is one component, and another of this same component is added in parallel structure until the entire budget is consumed. As an example;

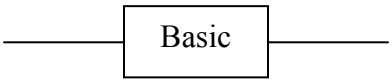


Figure 8 Basic Component with R_b = 0.6 and Cost $75

To increase R_b = 0.6 to R_p =0.9, the desired reliability level;

$$n = \left(\frac{C_s}{C_b}\right) \cong 3$$

This result shows that to increase basic system reliability from 0.6 to 0.9 requires 3 times basic cost. The new system reliability with three components in parallel structure is 0.936. Clearly, the aim of this formulation is to compute the budget needs to reach the desired reliability, not to maximize reliability under budget constraints, and this aim shows that this optimization method belongs to the first reliability optimization task mentioned above. In series structure, this formulation can be used to get the parallel-series structure. First, the reliability level of this series structure is computed, then by using formula in 18 the branch numbers are found. Here, each branch is accepted as one single component. This method is used for identical components, but in real world

problems, identical components are not always used, so this issue produces disadvantages for this method.

2.8.2 Series-Parallel Network (Homogeneous Case) Method.

The parallel systems method is the basic step for other methods; here the improved method for series-parallel systems will be examined. Dhillion discusses this method with following formulation and words.

A homogeneous case means that each element is identical with the same reliability and cost. This type of network is shown in figure 9. The basic system has k series elements. Each element reliability is given by $R_b^{1/k}$. We assume that system elements are statistically independent.

Suppose that one would like to increase the basic system reliability from R_b to R_{sp} [the final system (series-parallel network) reliability], which can be obtained by paralleling each basic element with an additional (n-1) elements. This will achieve the group of parallel elements (subsystem) reliability, R_{sp}.

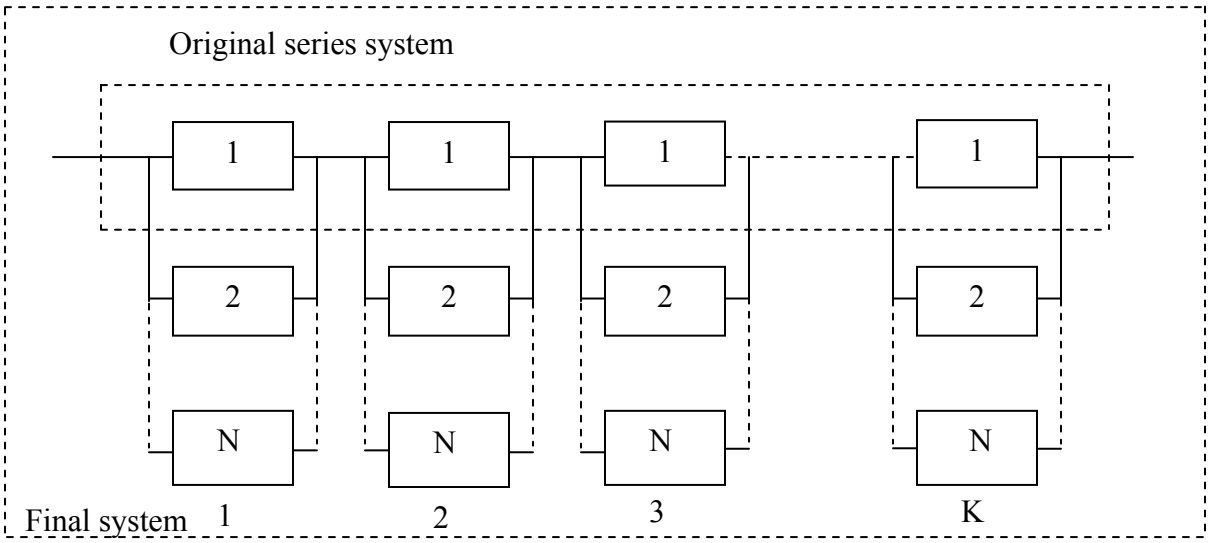


Figure 9 A Series-Parallel Network

Equation 19 is developed in the same way as eq.18

$$n = \left(\frac{C_s}{C_b}\right) = \frac{\log(1 - R_{sp}^{1/k})}{\log(1 - R_b^{1/k})} \qquad \textbf{(19)}$$

The above equation can be used to obtain the cost estimate if one desires to increase R_b to R_{sp}.

The formula in 19 is arranged for the identical components, and in real world problems, it is not easy to find systems with identical components. This formula accepts the equal number of components to be added into each subsystem, but this is not necessary for real problems, so this formula is not for practical usage most of the time. Meanwhile, the aim of this formulation, like parallel systems mentioned previously, is to compute the budget needs to reach the desired reliability, not to maximize reliability under the budget constraint. This method belongs to the first optimization task as well.

2.8.3 Series-Parallel Network (General Case) Method

A generalized model of the homogenous case is discussed in this section, and the assumption is that the reliability, R_i, and cost, C_i, of each individual element are different from others. An example of a basic series system with k non-identical elements can be seen in figure 10. Dhillion discusses this method with following formulation and words.

> Suppose the original cost and reliability of the figure 10 basic series system are C_b and R_b, respectively where the objective is to determine the minimum cost, C_s for increasing the basic series system reliability from R_b to R_{sp}. To increase reliability, we add the identical elements in parallel to each series element of figure 10.
>
> Assuming the system failures are statistically independent, the expressions

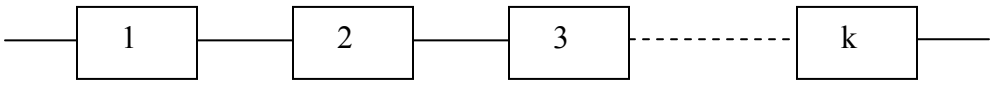


Figure 10 A Basic Series System

for this standard variational problem are developed as follows. The figure 10 basic system reliability and cost are given by:

$$R_b = \prod_{i=1}^{k} R_i \qquad \textbf{(20)}$$

and

$$C_b = \sum_{i=1}^{k} C_i \qquad \textbf{(21)}$$

If one assumes that the final system (series-parallel network) reliability is very large, then we could write

$$F_{sp} = 1 - R_{sp} << 1 \qquad \textbf{(22)}$$

where F_{sp} is the final system failure probability.

In order to obtain the specified system reliability, R_{sp}, parallel element 1, in figure 10 with n_1 identical elements; parallel element 2 with n_2 identical elements, and so on. Figure 11 shows the resulting overall configuration.

After paralleling the ith group (subsystem), the reliability is given by

$$R_{pi} = 1 - F_{pi}^{ni} \qquad \textbf{(23)}$$

where $F_{pi} = 1 - R_{pi}$

F_{pi} is the ith parallel group or subsystem element unreliability
n_i is the number of components or elements in the ith subsystem
Using equation 23, the final system reliability is

$$R_{sp} = \prod_{i=1}^{k} R_{pi} \qquad \textbf{(24)}$$

where k is the number of parallel subsystems in figure 11.

The ith parallel subsystem cost is $n_i C_i$, where C_i is the ith subsystem element cost. In other words, the $n_i C_i$ is the cost of ith group of parallel elements. The final system cost, C_s, is given by

$$C_s = \sum_{i=1}^{k} n_i C_i \qquad \textbf{(25)}$$

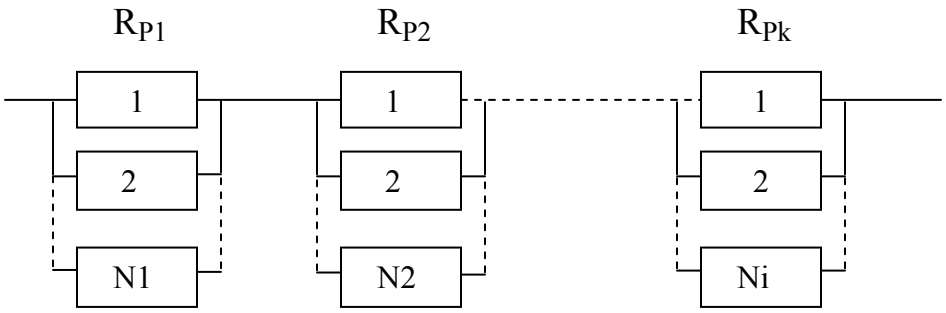


Figure 11 Improved Configuration of Figure 10; Increases Basic System Reliability

To find the minimum cost of a distribution of n_i 's solve equation 23 and 25 as a variational problem. This may not be as simple as it appears; therefore, it is desired to introduce variable, βi

$$R_{pi} = R_{sp}^{\beta i} \tag{26}$$

To satisfy equation 26 β_i has to be real, positive number between zero and unity. Utilizing equations 23 And 26, we get

$$n_i = \frac{\log(1 - R_{sp}^{\beta i})}{\log F_{pi}} \tag{27}$$

Substituting equation 27 in equation 25 results in

$$c_s = \sum_{i=1}^{k} (c_i \log(1 - R_{sp,\beta i})) / \log F_{pi} \tag{28}$$

To obtain the final reliability expression, substitute equation 26 in equation 24 to get

$$R_{sp} = (R_{sp})^{\sum_{i=1}^{k} \beta i} \tag{29}$$

Equation 29 is satisfied if the summation of β_i from i =1 to i = k is equal to unity. In other words, if

$$\sum_{i=1}^{k} \beta i = 1 \qquad \textbf{(30)}$$

The following expression for βi is derived in Moskowitz and McLean, "Some reliability aspects of system design," IRE Trans. On reliability and quality control, vol. RQc-8, September 1956, pp. 7-35;

$$\beta_i = \frac{c_i / \log F_{pi}}{\sum_{j=1}^{k} (c_j / \log F_{pj})} \qquad \textbf{(31)}$$

The aim of these formulations, like parallel systems and series-parallel systems with the homogeneous case mentioned above, is to compute the budget needs to reach the desired reliability, not to maximize reliability under budget constraint, so this method belongs to the first optimization task as well. Even though the general case accepts different components for each subsystem, these components are still identical for their own subsystems, and different brands cannot be added into those subsystems other than their own identical components. This is not practical in real world problems.

2.8.4 Steepest Descent Method.

This method is very practical, for series-parallel systems, but this method does not obtain the optimum solution every time. It can, however, be used for active and standby systems together. It demands some properties, for instance, "the cost, $C(X)$, is usually directly proportional to the number of identical units. The reliability function, $R(X)$, is usually a logarithmically convex additive function of its components $R1(X)$. Thus

$$\Delta(X) = R(X+1) - R(X) > 0$$

$$\Delta^2(X) = \Delta(X+1) - \Delta(X) < 0 \text{ for all x." (Ushakov, 1994)}$$

The algorithm has 6 steps and Ushakov discusses this issue with the following words and equations:

1) For the ith subsystem, a fixed interval of duration t_0, and for the alternative numbers of redundant units Xi, we compute the probability of failure free operation, $Pi(Xi)$. For convenience, we can put the data into a table.

REMARK: Here we consider the probability of failure-free operation during a given period of time, but we could instead consider an availability or operational availability coefficient.

2) We find the logarithm $Pi(Xi)$ values. The logarithm can be taken to any base.

3) On the basis of $Pi(Xi)$ and the known unit cost, c_i, we find values $g_i(Xi)$

$$g_i(Xi) = (\log Pi(Xi) - \log Pi(Xi-1))/c_i \qquad \textbf{(32)}$$

for all i and the proper number of units, X_i.
REMARK: For values $Pi(Xi)$ close to unity, the quantities $gi(Xi)$ can be approximated:

$$g_i(Xi) \approx (Pi(Xi) - Pi(Xi-1))/c_i \qquad \textbf{(33)}$$

4) All values $gi(Xi)$ are numbered in decreasing order.

5) We now consider a multiple-step procedure. First, choose

$$g^{(1)} = \{g_j^{(1)} : \max_{1\le i\le n} g_i^{(1)}\}$$

Compute

$$\log P^{(1)} = \log P^{(0)} - \log P_j(0) + \log P_j(1) \qquad \textbf{(34)}$$

Where

$$\log P^{(0)} = \sum_{1\le i\le n} \log P_i(0) \qquad \textbf{(35)}$$

is the initial value of the logarithm of the probability of system failure-free operation. Compute

$$C^{(1)} = C0 + c_j \quad \textbf{(36)}$$

where C_0 is the initial cost of the system.

Second, choose

$$g^{(2)} = \{g_k{}^{(1)} = \max_{1 \le i \le n} g_i{}^{(1)} \; for \; k \neq j \; , and \; g_j(2) \; for \; k = j\}$$

Find the corresponding $\log P_i(1)$ or $\log P_j(2)$. Compute

$$\log P^{(2)} = \log P^{(1)} - \log Pk(0) + \log Pk(1) \text{ or}$$

$$\log P^{(2)} = \log P^{(1)} - \log Pi(1) + \log Pi(2) \text{ compute,}$$

$$C^{(2)} = C^{(1)} + c_k \quad \textbf{(37)}$$

6) Terminate the process at step N:

$$N = \sum_{1 \le i \le n} xi \, , \quad \textbf{(38)}$$

where, for the minimum cost problem

$$\log P^{(N-1)} \le \log P_0 \le \log P^{(N)} \quad \textbf{(39)}$$

or, for the maximum effectiveness problem

$$C^{(N)} \le C_0 \le C^{(N+1)} \quad \textbf{(40)}$$

The steepest descent algorithm belongs to both the first and the second optimization tasks. One can use this method for both goals, try to maximize the system reliability under a certain budget constraint, or try to minimize budget while keeping at least a certain reliability level.

2.8.5 Marginal Analysis Method.

Marginal analysis deals with series-parallel systems, and it is a specific version of the steepest descent algorithm with the second optimization task and only active redundant systems. Ebeling discusses marginal analysis with following words:

> We will assume that the component reliabilities have been determined and that further improvement in system reliability is desired. The number of active redundant components is the relevant design variable. If a unit cost is associated with each component, an optimization scheme for allocating redundancy among the various components is possible. Assuming a serial relationship among a set of m independent components within a system, let
>
> $Ri(t)$ = (Known) reliability of component i at time t
>
> n_i = Number of parallel components i (decision variables)
>
> c_i = Unit cost of component i
>
> B = is budget available for additional units (redundancy)
>
> The problem is to find values for n_i so that
>
> $$\text{Max} \prod_{i=1}^{M}[1-(1-R_i(t))^{ni}] \qquad \textbf{(41)}$$
>
> Subject to:
>
> $$\sum_{i=1}^{m} c_i n_i \le B + \sum_{i=1}^{m} c_i$$
>
> The summation on the right side of the inequality accounts for the sunk costs necessary to have at least one of each component in the design. Marginal analysis may be used to solve this problem if the (natural) logarithm of the reliability function is maximized rather than the function itself. (Marginal analysis requires separability of the variables to ensure an optimal solution. If algorithms are used, the terms are additive rather than multiplicative. Since the algorithmic transformation is monotonically increasing, the optimal solution is not affected). Therefore the objective function becomes
>
> $$\text{Max} \sum_{i=1}^{m} \ln[1-(1-R_i(t))^{ni}] \qquad \textbf{(42)}$$

Eliminate the argument from $Ri(t)$, since the analysis is performed for a specified time t, and let

$$\Delta_i = \frac{\ln[1-(1-Ri)^{ni+1}] - \ln[1-(1-Ri)^{ni}]}{c_i} \qquad \textbf{(43)}$$

Then the marginal analysis consists of the following steps:

1) Set n_i = 1, i= 1,2…m, and set cost = 0
2) Compute Δ_i, i= 1,2…m
3) Find max $\{\Delta_1, \Delta_2 ... \Delta_m\}$; call it Δ_k
4) Set cost = cost + c_k
5) If cost < B, then set $n_k = n_k$ +1, recomputed Δ_k, and go to step 3; otherwise stop.

The marginal values Δi represents the increase in the algorithm of the component reliability per dollar investment in the ith component. At each iteration the component with the largest (current) marginal value is selected for an additional redundant unit. The process is repeated until the budget target is met. Since the budget may not precisely met by last component to be added, the final iteration may require selecting an alternate component having a smaller unit cost. The component having the largest marginal value with a unit cost that satisfy the budget is selected.

Ebeling gives an example of marginal analysis as follows:

An engineer has an $850 budget (per system) to be used to increase the reliability of a four-component series system. The reliability of each component at the desired system lifetime has been established through a reliability growth-testing program with following results:

Component	**Reliability**	**Unit Cost**
1	0.9	$100
2	0.85	$150
3	0.9	$50
4	0.95	$300

The objective is to maximize the system reliability through redundancy subject to the budget of $850.

Solution: The initial system reliability is 0.9*0.85*0.9*0.95 = 0.654.

Step 1: $n_1 = n_2 = n_3 = n_4 = 1$, and cost = 0.

Step 2: $\Delta_1 = 0.0009531$, $\Delta_2 = 0.000931746$, $\Delta_3 = 0.001906$, and $\Delta_4 = 0.00016263$.

Steps 3-5:

Table 1 The Steps Marginal Analysis Method

Iteration	Max Δ_i	k	Cumulative Cost	New Δ_k	System Reliability
1	0.001906	3	50	0.000181	0.719
2	0.0009531	1	150	0.0000905	0.791
3	0.00093175	2	300	0.0001292	0.91
4	0.000181	3	350	0.000018	0.918
5	0.00016263	4	650	0.0000079	0.964
6	0.0001292	2	800	0.0000192	0.983

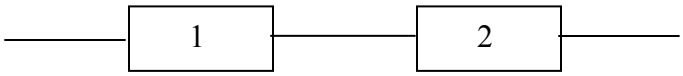
The next component to be included is component 1. However, the unit cost will result in the budget being exceeded. The only component that will keep the cost within budget is component 3. Therefore, the final solution is $n_1 = 2$, $n_2 = 3$, $n_3 = 4$, and $n_4 = 2$. The final system reliability is

$$R = [1- (1-0.9)^2][1- (1-0.85)^3][1- (1-0.9)^4][1- (1-0.95)^2]$$

This last step resulted in a very small increase in reliability for the dollars invested.

Because of the decreasing returns, it may be desirable to terminate the algorithm before the budget is reached. Although the above problem maximized reliability subject to a cost constraint, the constraint could have just easily been a weight or volume constraint.

Marginal analysis belongs to the second optimization task; it tries to maximize system reliability under certain budget constraints. Sometimes it produces mistakes. The following example shows how marginal analysis sometimes misdirects the solution after one iteration:



Reliability:	0.4	0.6
Cost:	$1.45	$1

Figure 12 Example of marginal analysis

If total budget is ignored for only one iteration, the marginal analysis will find, $\Delta_1 = 0.32414$, and $\Delta_2 = 0.33647$ from equation 43.

Marginal analysis will select the second subsystem, and this gives the system reliability as 0.336, but if the first subsystem were selected, it would give the system reliability as 0.384 after ending iteration one. Marginal analysis looks to accept identical components in each subsystem to be added, but in reality, it can be generalized for non-identical components without changing its algorithm.

Up to now, reliability allocation methods related with the scope of this research were given. To use these methods efficiently and fast, the computer codes are needed. The new heuristic, which is designed in this research, needs computer codes as well. Therefore, VBA programming language will be explained, and later it will be used for coding of new heuristic.

2.9 Visual Basic for Application (VBA)

VBA is Excel's programming language, and it has interaction with Excel's workbooks, sheets, which have 256 columns and 65536 rows, and charts. VBE, visual basic editor, is the place to write the code in modules. VBA code can read the data from

sheets, write the results on sheets, and draw graphs on chart sheets in Excel. VBA modules are saved in Excel workbooks, and one can shape the worksheets based on a special type of problem. VBA is an object-oriented programming language. Each object, such as cell or range, combination of cells, has some properties like name or value, and some methods can be applied to these objects like clear contents or copy.

"The VBA language consists of a backbone programming language, with typical programming elements you find in all programming languages: looping, logical if-then-else constructions, arrays, subroutines, variable types, and others."(Albright, 2001).

2.10 Summary

In this chapter, several issues were examined. First of all, heuristics are discussed because one of the objectives of this research is to find a new solution heuristic. Later, engineering design and design for reliability were examined so that the importance of reliability can be perceived effectively. Then, reliability block diagrams were examined, such as series systems and parallel systems, so that one can understand the nature of the connections of components in subsystems. In this way the design engineer can predict the reliability of systems with mathematical calculations. Following these, reliability optimization tasks and reliability improvement methods were discussed. In these sections, the main concern of this research is given in terms of tasks and improvement methods. Later, the known optimization methods are examined in order of simplicity. Finally, VBA was given because they will be used in the following chapters. To learn a programming language takes much time, therefore, VBA is mentioned only in a very short section.

III. Methodology

3.1 Introduction

Chapter two covered the background of reliability block diagrams and optimization methods in literature. This chapter provides a new heuristic. The reason for new heuristic is the lack of optimization tools in this area. Design engineer cannot use the common optimization tool LINGO because it can cope with only limited integer variables. Moreover LINGO cannot solve k-out-of-n systems because they require combination and summation terms in the objective function. The mentioned reasons are essential requirements of the new heuristic technique and VBA coding of this new technique. Using VBA and EXCEL, the design engineer can cope with many integer variables and complicated systems, even with k-out-of-n systems.

The new heuristic is based on multiplication logic. This is a new approach for solving reliability optimization problems. This logic will be explained in a detailed manner in the later sections. Remaining sections will examine terms, assumptions, definition of problem, general methodology, multiplication logic for larger results, understanding the new heuristic, steps of the new heuristic, simple example of the new heuristic, additional example with parallel and k-out-of-n systems, step-by-step of the overall algorithm, and a summary section.

3.2 Terms

This section will give definitions of terms that will be used in this research.

n: The number of subsystems in the system.

CS_i : Candidate set for subsystem i: This is the set of component available for each subsystem. A (p× 2) matrix, CS_i , defines the candidate set for each subsystem, where i = 1…n and p is equal to the number of components in candidate set i. The row number of CS_i identifies the candidate component number for this subsystem, the first column shows the reliability value of these components, and the second column shows the cost values of these components.

SCN: Subsystem candidate number is a vector, which defines how many components are in the subsystems candidate sets. This is also equal to ($|CS_1|$, $|CS_2|\ldots|CS_n|$). For instance, SCN = (3, 5…6) means that the first subsystem has 3 components in its candidate set, the second subsystems has 5 components in its candidate list, and so on. A special case of the SCN vector would be (1, 1... 1) where all subsystems have 1 component in their candidate sets. This is referred to as the Basic Component System (BCS).

K: This represents if the subsystem is pure parallel or k-out-of-n: F-system, which has different reliability calculation formula. If the elements of the K vector equal one, this means this subsystem has pure parallel redundancy. If K equals two or more than two, this means to work the subsystem needs at least K components. A vector shows K, and the elements of this vector belong to subsystems respectively. For instance,

K = (1, 2, 1, …1) has n elements, and K value of the first subsystem is 1, and 2 for the second subsystem, etc. This vector identifies the minimum number of components that must be operational for that subsystem to be operational.

N: This represents the desired (or minimum) component number in the initial system. For instance, if the subsystem is pure parallel, then N can be any number greater than or equal to one. But, if the subsystem is k-out-of-n: F-system, then N has to be greater than or equal to K. N is a vector and the elements of this vector belong to subsystems respectively. There are n elements in the vector. For instance, N = (1, 3, 2, …1) shows that the first subsystem has to have 1 component in the initial system design, and the second subsystem has to have 3, etc.

ISD: Initial system design: The initial system design is constructed according to system requirements. For instance, one subsystem is desired to have two components initially, or another subsystem is desired only one component initially. Initial system design is a function of N, K, n, and it can be drawn based on these values.

IS: Initial solution: This is the solution found by computing the initial system reliability after populating the initial system design with the proper components from the subsystem candidate sets. Then redundancy is implemented to improve the system reliability level.

MBR: Marginal benefit ratio: This ratio is computed for each component by the following formula:

MBR = Reliability of component / Cost of component

This ratio will be used to improve the new heuristic in the later sections. After explanation of terms, the assumptions of this research will be given in the next section.

3.3 Assumptions

Following assumptions will be accepted for this research:

1) All components are in two-state conditions. The components are in working condition or failure condition. This means the reliability of any component gives the working probability, and the unreliability of any component gives the failure probability.
2) The reliability and cost values of components in the basic system are constant at any time in the system life. Alternative ones are also to be added into the basic system.

3) The redundancy is accepted as active redundancy, which shares the load of the system at the same time.

4) The more reliable a component is, the more money it costs; otherwise the computation is not necessary, and more reliable components with the cheapest cost can be added into the basic system.

5) The system is in n-stage series structure. This means that the system has N subsystems in series structure at any time while it works, but the components in the subsystems are in the parallel structure.

6) Failures of components are independent of each other.

7) K-out-of-n systems use identical components.

The problem is how to construct the initial system design based on system requirements and optimize component design with active redundancy to obtain maximum reliability under budget constraints. This research will try to find a new greedy construction heuristic.

3.4 Definition of Problem

Selecting proper components from candidate sets for each subsystem sets up an n-stage series system. First, the initial system design will be set up, next the appropriate number of redundant components will be added to subsystems without exceeding the budget constraints. The new heuristic will be improved to accomplish this goal. After this short description of problem the general methodology will be given, and the picture of problem will be drawn.

3.5 General Methodology

The problem basically has three elements, the first thing is the initial system design, the second thing is candidate sets, and the last thing is budget. If the initial system design parameters, N, K, n, are given, the reliability block diagram is easily drawn. Then proper components are selected from candidate sets, which are shown by CS_i matrix, for subsystems according to initial system design and heuristic technique is implemented without exceeding the budget. For instance, if the initial system parameters are given as n = 5, K = (1, 2, 2, 1, 1), and N = (1, 2, 3, 2, 2). The initial system design is drawn based on these parameters, and shown in figure 13.

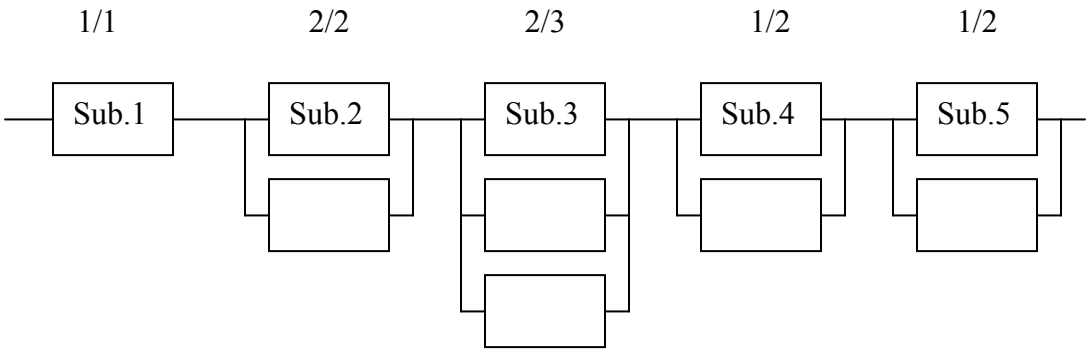


Figure 13 An Example of Initial System Design

After drawing the initial system design, which components will be put in the system has to be determined. All subsystems have their own candidate list, and the initial system design can be populated by the following three logical methods.

Best marginal benefit ratio (BMBR): The marginal benefit ratios are computed in each subsystem candidate set, then the largest one is selected to put into the initial system

design for this subsystem. The number of components can be more than one according to initial system design.

Maximum reliability (MR): The component with the maximum reliability value is selected from each candidate set; then it is put into the initial system design for this subsystem. The number of components can be more than one according to initial system design.

Minimum cost (MC): The component with the minimum cost is selected from each candidate set; then it is put into the initial system design for this subsystem. The number of components can be more than one according to initial system design.

After populating initial system design, new components are added to establish redundancy and improve the overall system reliability without exceeding the budget constraint. Therefore, the following section explains the new heuristic based on multiplication logic for adding redundant components.

3.5.1 Multiplication Logic for Larger Results.

Two numbers are assumed between 0 and 1, for instance, 0.7 and 0.9. The result of their multiplication is 0.63. If one change is allowed to improve the result, what should be done? Even if the second element, 0.9, were increased to 1, the maximum multiplication result would be 0.7. But, if the first element, 0.7, were increased to 0.78 the result would be 0.702. This value exceeds the maximum value of .7 when increasing the second component to a reliability of 1. Clearly, to improve the first element requires less effort to get the biggest result. Therefore, to obtain the biggest result, the lowest value should be increased as much as possible. The new heuristic, which is explained in this section, is based on this multiplication logic.

This section is discussed with the following example shown in figure 14, and the budget constraint is ignored.

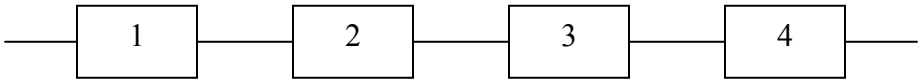


Figure 14 An Example Problem for Multiplication Logic

The reliability values of these four subsystems are 0.85, 0.87, 0.89, and 0.91, respectively. The system reliability is 0.5989. If the subsystem reliability values were changed to the reliability value of 1 sequentially while holding all other values at their original reliability values, then the maximum system reliability values would be 0.704613, 0.688415, 0.672945, 0.658155, respectively. If the first subsystem reliability value cannot be reached to 1, the system reliability value can be less than that of the second subsystem produces when it reaches to 1. This is called misdirection of multiplication logic.

3.5.2 Understanding the New Heuristic.

The new heuristic is developed for redundancy problems. This means that the initial system design is accepted and is already constructed by any means of three approaches. These approaches, which are explained in general methodology section, are best marginal benefit ratio, maximum reliability, and minimum cost. A simple example is provided to assist the reader and understanding the heuristic. The parameters of the problem are n =2, K = (1, 1), N = (1, 1),

$$CS_1 = \begin{matrix} 0.6 & 60 \\ .65 & 67 \\ .75 & 72 \end{matrix}, \quad CS_2 = \begin{matrix} 0.7 & 70 \\ .75 & 73 \\ .8 & 82 \end{matrix}$$

therefore, the initial system design will be as follows:

k-out-of-n

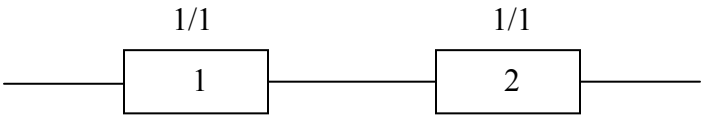


Figure 15 2-Stage Series System

Using the best marginal benefit ratio (BMBR), the initial system design (ISD) can be populated. All marginal benefit ratios have to be computed for each component. In subsystem 1, the first component has the ratio of 0.01 (0.6 / 60), the second component and the last component have the ratios of 0.0097 and 0.0104 respectively. The last component has the largest ratio, and when subsystem 1 is selected, the last component (R=0.75) will be added into this subsystem. The same ratio calculations are done for subsystem 2 and the second component from the candidate set (reliability equal to .75) is added to the subsystem. Therefore, the populate ISD is shown in figure 16.

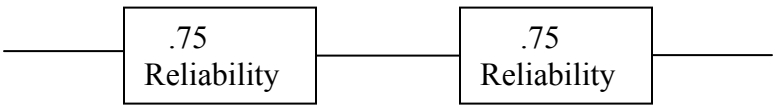


Figure 16 2-Stage Series System After Initial Design

Now, the subsystem redundancy can be increased based on the multiplication logic of the new heuristic.

If two or more subsystems have equal reliability (as in this example) at the same time, the component MBR values for these subsystems have to be evaluated. This means that the subsystem with the largest MBR value will be selected. For instance, the two subsystems in the example have equal reliability. The largest MBR values form their respective candidate sets are 0.0104 and 0.0103. Subsystem 1 is selected because it has the largest MBR value for its components. What if two or more components have equal marginal benefit ratios at the same time? Which one should be added into its own subsystem?

If one subsystem has equal MBR values, on the surface it looks as if any of the components can be selected, but this issue has to be researched deeply. This problem is illustrated with the following parameters; $CS_1 = \begin{matrix} 0.8 & 80 \\ 0.4 & 40 \\ 0.7 & 75 \end{matrix}$ and MBR values are 0.01, 0.01, 0.00933, respectively.

The first and second components have the same largest MBR values, 0.01, together. If the first component is selected, the new subsystem reliability will be 0.96, and the consumed budget is \$80. If the second component 1 is selected, then new subsystem reliability will be 0.88, and consumed budget is \$40. Here, there is \$40 remaining compared to the consumed budget by the basic component. Therefore, one more second component 1 can be added. After adding the second component, the new subsystem reliability will be 0.928, and consumed budget is \$80. Adding the first component gives larger new subsystem reliability, 0.96, than adding the second component, which produces the system reliability value of 0.928. Therefore, as a new

rule of this new heuristic, the component with larger reliability will be determined at the beginning of the solution if more than one component has equal ratio values.

3.5.3 Steps of the New Heuristic.

The overall steps to use the new heuristic are applied to n-stage series systems. The algorithm is as follows:

1) Determine the initial system design (ISD) by defining n, K, and N.
2) Populate the ISD by using one of the three approaches, BMBR, MR, or MC.
3) Determine the marginal benefit ratios of components in each candidate sets. Order the MBR values in descending order.

Note: If there is a tie between component ratios, select the component with the highest reliability level.

4) The solution at each iteration selects the subsystem with the lowest reliability value at that time. If more than one subsystem has equal reliability level at the same time, select the subsystem with the largest MBR value. If the largest components are the same with every value, arbitrarily choose the subsystems to improve.
5) Identify the top ranked component from the candidate set to include into the selected subsystem. Check the remaining budget to determine if this component can be added. If so, add this component to the subsystem and update the reliability value of this subsystem and the available budget. If the remaining budget is not enough for the component with the highest MBR, then try the second component in order. If this second one is not proper, continue to the third one and so on. If the candidate set is exhausted with no component selected due to budget, return to step 4.

Repeat steps 4 and 5 until the entire budget is consumed or the remaining budget is not enough for the components to be added.

Compute the overall system reliability value and provide the final system design (FSD).

3.5.4 Simple Example of the New Heuristic.

The following example is in Ebeling's book, 1997, and it considers only the basic components system (BCS) to improve the system reliability. The problem is shown in the figure 17. The problem parameters are n = 4, K = (1, 1, 1, 1), N = (1, 1, 1, 1), CS_1 = (0.9, 100), CS_2 = (0.85, 150), CS_3 = (0.9, 50), CS_1 = (0.95, 300), and extra budget is $850 after putting one components for each subsystem from their own candidate sets. The MBR values will be 0.009, 0.00567, 0.018, and 0.003167.

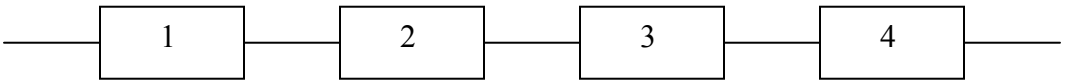


Figure 17 Example of New Heuristic

The initial system reliability is computed as 0.654075 (0.9 * 0.85 * 0.9 * 0.95). The table 1 shows which components are selected at the end of iterations by using the new heuristic.

Table 2 The Result of Example Question with New Heuristic

ITER.	SUBSYS.	SUBSYS. RELIABILITY	USED BUDGET	NEW SYS.RELI.
1	2	0.85	$150	0.752186
2	3	0.9	$150+50	0.827404
3	1	0.9	$200+100	0.910145
4	4	0.95	$300+300	0.9556526
5	2	0.9775	$600+150	0.97435

6	3	0.99	$750+50	0.982837
7	3	0.999	$800+50	0.98372

The final system reliability is 0.98372, which is the same result with the marginal analysis method, and was shown in chapter two. Except for the last iteration, the two heuristics selected different subsystems during the iterations steps. The marginal analysis selections at the end of each iteration were 3,1,2,3,4,2,3, while the new heuristic selected 2, 3, 1, 4, 2, 3, and 3. In the last two iterations, the new heuristic selected subsystem three because it is the only subsystem with the cost within the remaining budget. Overall results for the FSD are given in the following table 2:

Table 3 Overall Results of Example Question with New Heuristic

	ADDED COMP. NO	**USED BUDGET**	**TOTAL COMP. NO**
SUBSYSTEM 1	1	$100	2
SUBSYSTEM 2	2	$300	3
SUBSYSTEM 3	3	$150	4
SUBSYSTEM 4	4	$300	2

The entire budgets is consumed, $850, and the system reliability level is 0.98372.

3.5.5 Additional Example with Parallel and K-out-of-N Systems.

In real world problems, the basic system can include both pure parallel systems and k-out-of-n systems. The ISD is defined by n = 3, K = (1, 1, 2) and N = (2, 1, 3). The following figure 17 illustrates this situation:

k-out-of-n

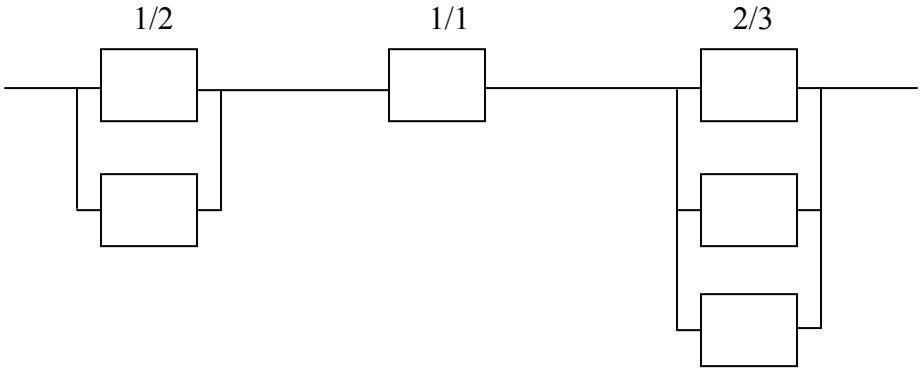


Figure 18 Basic system with parallel and k-out-of-n systems

The new heuristic will perform the same algorithm using the computed subsystem reliabilities. Again, the subsystem with the lowest reliability value will be selected at the end of iterations, and one new component will be added. Then the reliability level of this subsystem and available budget values will be updated, and so on. Note: There is the assumption (#7) for k-out-of-n systems that all components in the k-out-of-n systems are identical components. This assumption forces all components within a subsystem that is not pure parallel to be identical components. If the ISD is populated using the MR or MC approaches, these same initially selected components are the only potential candidate components as the algorithm builds redundancy into the system design..

The heuristic algorithm will demonstrate its importance here, because it can be used with k-out-of-n systems easily and thousands of integer variables at the same time with VBA coding. Conversely, the common optimization tool, LINGO, cannot be used with k-out-of-n systems.

3.5.6 Step-by-Step Example of the Overall Algorithm

In this section, the initial system design and new heuristic will be used to solve an example problem.

Step 1) The problem parameters are n = 3, K = (1, 1, 1), N = (1, 1, 1),

$$CS_1 = \begin{matrix} 0.6 & 100 \\ 0.65 & 120 \\ 0.9 & 145 \end{matrix} \quad CS_2 = \begin{matrix} 0.7 & 190 \\ 0.75 & 250 \\ 0.85 & 270 \end{matrix} \quad CS_3 = \begin{matrix} 0.9 & 300 \\ 0.95 & 310 \\ 0.97 & 315 \end{matrix}$$

the budget is $2000.

Step 2) The initial system design is populated according to three approaches and then the new heuristic is implemented to them.

Step 3) According to this data, the MBR values of subsystem one are 0.006, 0.0054, 0.0062, respectively. The MBR values of subsystem two are 0.00368, 0.003, 0.00314, respectively. The MBR values of subsystem three are 0.003, 0.00306, 0.00379, respectively.

Step 4-7) The new heuristic is implemented and the results are shown in table 3.

Table 4 The Results of Example Problem with Initial Design and New Heuristic

	Number of Components	System Reliability	Used Budget
Best MBR	**10**	**0.9909**	**$1,970**
Maximum Reliability	**9**	**0.99405**	**$1,905**
Minimum Cost	**11**	**0.99059**	**$1,955**

In this example, starting with components which have the maximum reliability values, produces the largest system reliability level after implementing the new heuristic; moreover, it uses the least budget. Of course, to make an inference by looking at one

example is impossible; so all three approaches will be used as the initial design. The new heuristic will be implemented using VBA coding.

3.7 Summary

This chapter mainly discussed terms, assumptions, initial system design approaches, multiplication logic for larger results, improvement of new heuristic and merging this new heuristic with initial design approaches and example problems. After defining the new heuristic, the next chapter will discuss coding the new heuristic together with marginal analysis heuristic, solve example problems to evaluate the performance of the new heuristic, and compare two heuristics.

IV. Results

4.1 Introduction

The last chapter examined the methodology of this research. GOZEBE will be used as the new heuristic. This methodology covered initial system design and found a new heuristic based on multiplication logic and marginal benefit ratios of components. This chapter addresses using the software of new heuristic together with initial system design parameters. Moreover, this coding will include the marginal analysis heuristic so that the design engineer can select the best solution from two heuristics. Meanwhile, both heuristics will be applied to three different initial design approaches explained in chapter three.

After explaining the software and how to use it, the remaining sections of this chapter will use the software with randomly produced examples to compare the new heuristic to marginal analysis heuristic. The statistical tests will be performed for this comparison.

4.2 The VBA Codes of Software

The VBA codes of software are given in appendix A. This software includes both marginal analysis heuristic and the new heuristic to provide redundancy after determining the initial system design. This design will be found with its parameters, n, K, N. The components to construct the initial system will be selected by using the three different approaches: maximum reliability, best marginal benefit ratio, and minimum cost. The software will automatically put the components into the initial system, and use two

different heuristics, which are the new heuristic and marginal analysis heuristic, to provide redundancy so that the system reliability value is improved. The next section will explain how to use the software and give illustrations.

4.2.1 How to Use The New Software.

The new software includes two EXCEL books; the first one is the ReliabilityHeuristicMenu book, the other one is the ProblemDataParamSolution book. The Menu book is used for main operations such as opening, saving a book, exiting from the book, and running the heuristic after opening the ProblemDataParamSolution book. The view of this main book is in the figure 19.

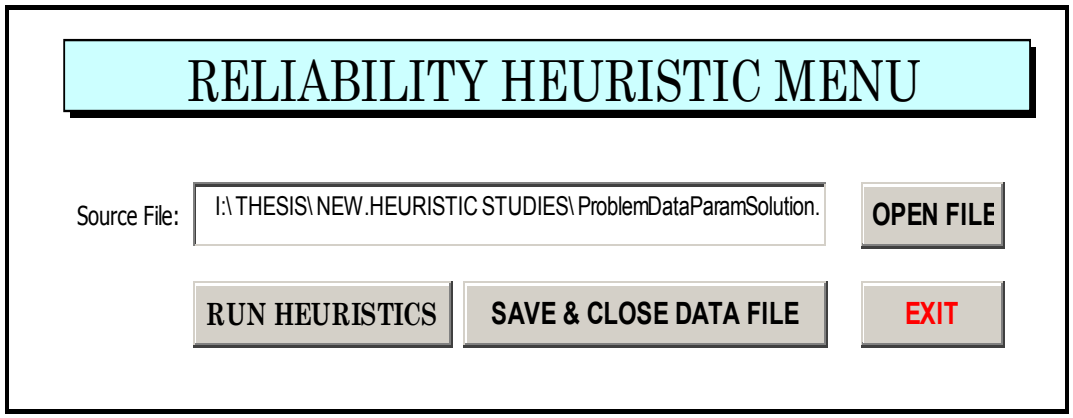


Figure 19 The View of Main Book of Software

The definitions of bars in the book are as follows.

OPEN FILE: This bar is used to open the ProblemDataParamSolution book to write the problem data such as candidate sets, parameters such as initial system design parameters, and budget.

RUN HEURISTICS: This bar is used to populate the initial system design and provide redundancy through both heuristics.

SAVE & CLOSE DATA FILE: This bar is used to save the ProblemDataParamSolution book.

EXIT: This bar is used to exit the ReliabilityHeuristicMenu book.

After explaining the first book, the second book will be explained. This book is for problem data, parameters, budget, and solutions. This book has five worksheets. These are Data, Initial Solution, Solution-Ratio, Solution-Reliability, and Solution-Cost. The following figures and their related worksheets are described in table 5 to provide an overview of this EXCEL spreadsheet.

Table 5 The Quick View of Worksheet Figures

STATUS	FIGURE NO	FIGURE NAME	WORKSHEET
INPUT	19	The View of Data Worksheet	DATA
OUTPUT	20, 21	The View of the Initial Solution Worksheet	INITIAL SOLUTION
	22, 23, 24, 25	The View of the Solution-Ratio Worksheet	SOLUTION-RATIO
	26, 27, 28, 29	The View of the Solution-Reliability Worksheet	SOLUTION-RELIABILITY
	30, 31, 32, 33	The View of the Solution-Cost Worksheet	SOLUTION-COST

The view of the data sheet is in figure 20.

Component	Reliability	Cost	SubSystem
1	0.800	$80.0	1
2	0.900	$87.0	1
3	0.875	$90.0	2
4	0.925	$95.0	2

SubSystem	K-Vector	N-Vector
1	1	1
2	2	2

Total Budget
$540

Figure 20 The View of Data Worksheet

The Data worksheet has three charts to fill in. The first chart is to input candidate sets. The first column of this chart is the component column. The components are numbered without interruption. This means that the component numbers start with the first subsystem candidate set, and the first component takes number one, the second number two, and so on. Therefore, the last component number of the last subsystem shows how many total components this problem has. The second chart is to input K and N vectors. The third chart is to input the budget value.

The second worksheet is for the initial solution. After filling the Data worksheet, the user has to save this excel workbook by giving it a new name or using its original name, then closing this book. To run the heuristics with this data, the user has to open this workbook by using the ReliabilityHeuristicMenu workbook OPEN bar, and hitting the RUN HEURISTICS bar. After a short time the computations will be done. The view of the Initial Solution worksheet is in figure 21 and figure 22.

BEST MARGINAL BENEFIT RATIO					
				System Reliability	System Cost
				0.77006	$277.0
SubSys	Comp	k	N	Reliability	Cost
1	2	1	1	0.90000	$87.0
2	4	2	2	0.85563	$190.0

MAXIMUM RELIABILITY					
				System Reliability	System Cost
				0.77006	$277.0
SubSys	Comp	k	N	Reliability	Cost
1	2	1	1	0.90000	$87.0
2	4	2	2	0.85563	$190.0

Figure 21 The View of the Initial Solution Worksheet

CHEAPEST COST					
				System Reliability	System Cost
				0.61250	$260.0
SubSys	Comp	k	N	Reliability	Cost
1	1	1	1	0.80000	$80.0
2	3	2	2	0.76563	$180.0

Figure 22 The View of the Initial Solution Worksheet

The charts in figure 21 and 22 are next to each other in the actual worksheet; there is only one EXCEL column between them. The first chart shows the initial solution if the components with the best MBR values in each subsystem are put into the system. The value under "System Reliability" in the fifth column gives the initial system reliability value. The value under "System Cost" in the sixth column represents the initial system cost. The second chart shows the results if the components with maximum reliability values are put in the system. The third chart gives the results if the components with cheapest cost are put in the system.

The remaining three worksheets (Solution-Ratio, Solution-Reliability, Solution-Cost Worksheets) will show the results after implementing both heuristics to initial system designs. The Solution-Ratio worksheet is based on the best marginal benefit ratio initial design. The heuristics are applied to this initial system value. Figure 23 and figure 24 illustrate the iterations to solve this problem.

GOZEBE - BEST RATIO							
						System Reliability	System Cost
						0.98200	$539.0
Status	Step	SubSys	Comp	k	N	Reliability	Cost
Initial	1	1	2	1	1	0.90000	$87.0
Initial	2	2	4	2	2	0.85563	$190.0
1	1	2	4	-	-	0.92500	$95.0
1	2	1	2	-	-	0.90000	$87.0
2	1	1	1	-	-	0.80000	$80.0
Final	1	1	-	-	-	0.99800	$254.0
Final	2	2	-	-	-	0.98397	$285.0

Figure 23 The View of the Solution-Ratio Worksheet

MARGINAL ANALYSIS - BEST RATIO							
						System Reliability	System Cost
						0.98200	$539.0
Status	Step	SubSys	Comp	k	N	Reliability	Cost
Initial	1	1	2	1	1	0.90000	$87.0
Initial	2	2	4	2	2	0.85563	$190.0
1	1	2	4	-	-	0.92500	$95.0
1	2	1	2	-	-	0.90000	$87.0
2	1	1	1	-	-	0.80000	$80.0
Final	1	1	-	-	-	0.99800	$254.0
Final	2	2	-	-	-	0.98397	$285.0

Figure 24 The View of the Solution-Ratio Worksheet

These two charts show the results after implementing both heuristics. The Status column shows the situation of the solution. For instance, the term “initial” in the status column shows initial solution values. Number one means the heuristics use the best proper components at this time; number two uses second best components. The term final shows final subsystem values after every thing is done. The best proper component numbers in the previous sentence mean that the software puts the component with the largest MBR value for GOZEBE heuristic. If budget is not available for this component,

the software uses the second best component and so on. The Step column shows the order of selecting subsystems. The SubSys column shows which subsystem is selected at this step. The Comp column shows which component is put at the end of this step. The value under "System Reliability" in the fifth column gives the initial system reliability value, and it is computed by multiplying final row values under System Reliability column. The value under "System Cost" in the sixth column represents the initial system cost, and it is computed by adding the final row values under the System Cost column. The summary results of GOZEBE heuristic are illustrated in figure 25.

GOZEBE-BEST RATIO-SUMMARY				
		Total Components	System Reliability	System Cost
		6	0.98200	$539.0
SubSys	Comp	Quantity	Reliability	Cost
1	1	1	0.80000	$80.0
1	2	2	0.90000	$87.0
2	4	3	0.92500	$95.0

Figure 25 The View of the Solution-Ratio Worksheet

The first column shows the subsystem number; the second column shows the number of components added into this subsystem; and the third column shows the quantity of components. The total quantity is under "Total Components". The fourth column shows the individual reliability values of these components, and the overall system reliability value is under the "System Reliability" cell after putting this amount of components into the system. The last column shows the individual cost values of these components, and the overall system cost value is under the "System Cost" cell after putting this amount of components into the system. The summary results of marginal

analysis heuristic are illustrated in figure 26, and it has the same explanation as the previous explanation, therefore only the figure will be given.

MA-BEST RATIO-SUMMARY				
		Total Components	System Reliability	System Cost
		6	0.98200	$539.0
SubSys	Comp	Quantity	Reliability	Cost
1	1	1	0.80000	$80.0
1	2	2	0.90000	$87.0
2	4	3	0.92500	$95.0

Figure 26 The View of the Solution-Ratio Worksheet

The Solution-Reliability worksheet has the same illustration and explanation. The only difference is that this workbook starts the computation with the initial system made up of components with the maximum reliability value without caring about their costs.

Therefore, only their illustrations will be given in the figure 27, figure 28, figure 29, and figure 30.

GOZEBE - MAXIMUM RELIABILITY							
						System Reliability	System Cost
						0.98200	$539.0
Status	Step	SubSys	Comp	k	N	Reliability	Cost
Initial	1	1	2	1	1	0.90000	$87.0
Initial	2	2	4	2	2	0.85563	$190.0
1	1	2	4	-	-	0.92500	$95.0
1	2	1	2	-	-	0.90000	$87.0
2	1	1	1	-	-	0.80000	$80.0
Final	1	1	-	-	-	0.99800	$254.0
Final	2	2	-	-	-	0.98397	$285.0

Figure 27 The View of the Solution-Reliability Worksheet

MARGINAL ANALYSIS - MAXIMUM RELIABILITY							
						System Reliability	System Cost
						0.98200	$539.0
Status	Step	SubSys	Comp	k	N	Reliability	Cost
Initial	1	1	2	1	1	0.90000	$87.0
Initial	2	2	4	2	2	0.85563	$190.0
1	1	2	4	-	-	0.92500	$95.0
1	2	1	2	-	-	0.90000	$87.0
2	1	1	1	-	-	0.80000	$80.0
Final	1	1	-	-	-	0.99800	$254.0
Final	2	2	-	-	-	0.98397	$285.0

Figure 28 The View of the Solution-Reliability Worksheet

GOZEBE-MAXIMUM RELIABILITY-SUMMARY				
		Total Components	System Reliability	System Cost
		6	0.98200	$539.0
SubSys	Comp	Quantity	Reliability	Cost
1	1	1	0.80000	$80.0
1	2	2	0.90000	$87.0
2	4	3	0.92500	$95.0

Figure 29 The View of the Solution-Reliability Worksheet

MA-MAXIMUM RELIABILITY-SUMMARY				
		Total Components	System Reliability	System Cost
		6	0.98200	$539.0
SubSys	Comp	Quantity	Reliability	Cost
1	1	1	0.80000	$80.0
1	2	2	0.90000	$87.0
2	4	3	0.92500	$95.0

Figure 30 The View of the Solution-Reliability Worksheet

The Solution-Cost workbook has the same logic as the previous workbooks. The only difference is that it starts the computation with the initial system made up of components with the cheapest cost value without caring about their reliability values. Therefore, only their illustrations will be given in figure 31, figure 32, figure 33, and figure 34.

GOZEBE - CHEAPEST COST							
						System Reliability	System Cost
						0.97306	$527.0
Status	Step	SubSys	Comp	k	N	Reliability	Cost
Initial	1	1	1	1	1	0.80000	$80.0
Initial	2	2	3	2	2	0.76563	$180.0
1	1	2	3	-	-	0.87500	$90.0
1	2	1	2	-	-	0.90000	$87.0
1	3	2	3	-	-	0.87500	$90.0
Final	1	1	-	-	-	0.98000	$167.0
Final	2	2	-	-	-	0.99292	$360.0

Figure 31 The View of the Solution-Cost Worksheet

MARGINAL ANALYSIS - CHEAPEST COST							
						System Reliability	System Cost
						0.97306	$527.0
Status	Step	SubSys	Comp	k	N	Reliability	Cost
Initial	1	1	1	1	1	0.80000	$80.0
Initial	2	2	3	2	2	0.76563	$180.0
1	1	2	3	-	-	0.87500	$90.0
1	2	1	2	-	-	0.90000	$87.0
1	3	2	3	-	-	0.87500	$90.0
Final	1	1	-	-	-	0.98000	$167.0
Final	2	2	-	-	-	0.99292	$360.0

Figure 32 The View of the Solution-Cost Worksheet

GOZEBE-CHEAPEST COST-SUMMARY				
		Total Components	System Reliability	System Cost
		6	0.97306	$527.0
SubSys	Comp	Quantity	Reliability	Cost
1	1	1	0.80000	$80.0
1	2	1	0.90000	$87.0
2	3	4	0.87500	$90.0

Figure 33 The View of the Solution-Cost Worksheet

MA-CHEAPEST COST-SUMMARY				
		Total Components	System Reliability	System Cost
		6	0.97306	$527.0
SubSys	Comp	Quantity	Reliability	Cost
1	1	1	0.80000	$80.0
1	2	1	0.90000	$87.0
2	3	4	0.87500	$90.0

Figure 34 The View of the Solution-Cost Worksheet

4.3 Comparing the Two Heuristics with Randomly Produced Examples

The reliability books in literature generally examine systems with 4 or 5 subsystems. This section will enlarge the problem size and examine these larger systems. The initial parameters of these systems are n = 10 and a variety of N and K vectors. Because, there is no limit to use marginal analysis heuristic for any basic system, marginal analysis will be used to provide redundancy on 3 initial system design populated by the new heuristic. Therefore, both heuristics (GOZEBE & M.A.) are used in the result analysis.

The aim of the design engineer is to obtain a system that has the reliability value near 1. The cost values of the components are accepted to range from $500 to $1000. The system budget will be accepted as $30,000 for the systems with n = 10 subsystems.

There are four categories of examples. The first category examined one hundred randomly produced examples, and these examples include only pure parallel, 1-out-of-N, subsystems. The problem parameters of these 100 examples are n = 10, K = (1, 1, 1, 1, 1, 1, 1, 1, 1, 1), N = (1, 1, 1, 1, 1, 1, 1, 1, 1, 1), SCN = (4, 4, 4, 4, 4, 4, 4, 4, 4, 4), and budget is $30,000. The second category accepted one 2-out-of-N subsystem and nine pure parallel subsystems while n, SCN, and budget are the same. The third category included two 2-out-of-N subsystems and eight pure parallel subsystems while n, SCN, and budget

are the same. The last category include ten 2-out-of-N subsystems with $40,000 budget while n, SCN are the same. The reason for the budget increase is the k-out-of-n system produces less reliability value than pure parallel systems if the same components of equal number are used. The entire results are in appendix B, C, D, and E.

4.3.1 General Evaluation of Example Problems.

First, the results based on reliability values will be evaluated. The overall number of being first based on reliability is given in the table.

Table 6 Summary Results of Being First

NUMBER OF WINNERS BASED ON RELIABILITY					
	Pure parallel	K. No:1	K. No:2	K. No:10	TOTAL
GOZEBE	35	12	13	9	69
MARGINAL ANALYSIS	18	11	12	10	51
TIES	47	27	25	31	130
TOTAL	100	50	50	50	250

In the first category, the new heuristic had 35 larger reliability values while marginal analysis had 18 larger reliability values. In other categories, they have almost the same numbers. It can be inferred that GOZEBE results in a larger reliability. Therefore, the sign test will be run for pure parallel example category and total results.

For the pure parallel results, our n = 53 for sign test, 53 is the total number of systems with a clear winner. Ties are discarded. Because n is greater than 25, the normality is approximated. The test result is in the table 7.

Table 7 Sign Test Result For Pure Parallel Examples Category Based on Realiability

SIGN TEST FOR PURE PARALLEL CATEGORY	
H0: p = 0.5	
H1: p > 0.5	
alpha =	0.05
M =	35
Z =	2.34
Z-alpha =	1.644
RESULT:	REJECT H0 HYPOTHESIS
p-value =	0.010

There is statistical difference at alpha, 0.05, the p value, 0.01. This means the new heuristic is better. The sign test for the total examples is given in table 8.

Table 8 Sign Test Result For Total Examples Based on Reliability

SIGN TEST FOR TOTAL EXAMPLES	
H0: p = 0.5	
H1: p > 0.5	
alpha =	0.05
M =	69
Z =	1.643
Z-alpha =	1.645
RESULT:	ACCEPTS H0 HYPOTHESIS
p-value =	0.050

Therefore, there is no statistical difference between two heuristics based on the number of being first on reliability. Therefore, for pure parallel GOZEBE performs better while for overall reliability results there is no statistical difference for alpha=0.05.

The number of components used by heuristics is very important for design engineers because using more components makes the system more complex. Moreover, it makes the system heavy. Especially, in aerial vehicles and satellites, it is a desired thing to be less heavy. The following table shows the more component usage by heuristics.

Table 9 Number of Examples Based on More Component Usage for Heuristics

NUMBER OF EXAMPLES BASED ON MORE COMPONENT USAGE					
	Pure parallel	K. No:1	K. No:2	K. No:10	TOTAL
GOZEBE	0	0	1	0	1
MARGINAL ANALYSIS	19	11	10	4	44
TIES	81	39	39	46	205
TOTAL	100	50	50	50	250

In the first example category, marginal analysis used more components in 19 examples while GOZEBE used more componets in zero examples. In third example category, GOZEBE used more components only one time because of starting with a different initial system. The sign test will be done for K.No: 10, K.No: 2, and total results.

Table 10 Sign Test Result for The Fourth Category with K.No = 10

SIGN TEST FOR K.No=10 EXAMPLES		
H0: p = 0.5		
H1: p > 0.5		
alpha =	0.0625	= P(4)
M =	4	
RESULT:	REJECT H0 HYPOTHESIS	

Therefore, we conclude that sufficient evidence exists to indicate that the two heuristic are different based on component usage while K.No = 10. This means the new heuristic better than marginal analysis heuristic based on component usage.

Table 11 Sign Test Result for The Third Category with K.No = 2

SIGN TEST FOR K.No=2 EXAMPLES		
H0: p = 0.5		
H1: p > 0.5		
alpha =	0.0327	= P(11)+P(10)+P(9)
M =	10	
RESULT:	REJECT H0 HYPOTHESIS	

Therefore, we conclude that sufficient evidence exists to indicate that the two heuristic are different based on component usage while K.No = 2. There is no need to run sign test for K.No = 1, and pure parallel subsystems example categories, because the number for GOZEBE is greater than that of K.No = 2, and the number for marginal analysis is less than that of K.No = 2. The sign test result for the total evaluation is given in the following table. Because n >= 25, normality assumption can be made.

Table 12 Sign Test Result for Total Example Results Based on More Comp. Usage

SIGN TEST FOR TOTAL EXAMPLES	
H0: p = 0.5	
H1: p > 0.5	
alpha =	0.05
M =	44
Z =	6.41
Z-alpha =	1.645
RESULT:	REJECT H0 HYPOTHESIS
p-value =	0.00000

Therefore, under all categories and in total evaluation, the new heuristic is better than marginal analysis based on component usage. This feature provides the new heuristic priority to marginal analysis.

4.3.2 Evaluating the Initial System Design Approaches.

First, the initial system design approaches are examined. Table 13 shows which design approaches resulted in the highest reliability. The number before "/" is for GOZEBE heuristic; the number after "/" is for marginal analysis heuristic.

Table 13 General Evaluation of Initial Design Approaches Based on Reliability

GENERAL EVALUATION OF INITIAL DESIGN APPROACHES				
	Pure parallel	K. No:1	K. No:2	K. No:10
Best MBR	2 / 2	2 / 3	5 / 4	11 / 11
Maximum Reliability	98 / 98	48 / 47	45 / 46	39 / 39
Minimum Cost	0	0	0	0

In the pure parallel subsystems example category, we see that 98 examples used maximum reliability initial design approach, and 2 examples used best MBR initial design approach. Minimum cost initial design approach was not used by any example in any category. It can be inferred that there is no relation between initial design approaches and heuristics because the numbers before and after slashes are almost the same.

The sign test will be implemented to see whether or not there is statistical difference between BMBR and maximum reliability intial design approaches. First, the sign test will be done for K.No = 10. Because n >= 25, normality assumption can be made .The following table gives the test result.

Table 14 Sign Test Result for Initial Desgin Approaches Based on Reliability

SIGN TEST FOR INTIAL DESIGNS	
H0: p = 0.5	
H1: p > 0.5	
alpha =	0.05
M =	39
Z =	3.96
Z-alpha =	1.645
RESULT:	REJECT H0 HYPOTHESIS
p-value =	0.00004

Clearly, maximum reliability initial design approach is the best of the others based on reliability values. The sign test is not than for other categories, because the

numbers in this test are more advantageous to maximum reliability intial design approach than that of K.No = 10. Hence, it is better in all categories.

The following table shows in how many examples the less component is used, these results are computed after finishing the whole design process and they are based on intial design approaches. For this table, the first 50 examples of the first example category are used.

Table 15 Numbers of Examples For Initial Design Approaches Based on Less Component Usage

NUMBER OF EXAMPLES BASED ON LESS COMPONENT USAGE			
	BMBR	MAX. RELIABILITY	MIN. COST
# of EXAMPLES	0	50	0

Clearly, the maximum reliability approach used fewer components in all examples. The sign is made for only BMBR and maximum reliability approach because BMBR and minimum cost initial design approaches have the same number, 0. The test result in the following table.

Table 16 Sign Test Result for Initial Desgin Approaches Based on Less Comp. Usage

SIGN TEST FOR INTIAL DESIGNS	
H0: p = 0.5	
H1: p > 0.5	
alpha =	0.05
M =	50
Z =	7.07
Z-alpha =	1.645
RESULT:	REJECT H0 HYPOTHESIS
p-value =	0.00000

Clearly, maximum reliability initial design approach is the best of the three approaches.

4.3.3 Comparing GOZEBE Heuristic with LINGO Optimization Tool

Because LINGO cannot solve systems involving k-out-of-n subsystems, only systems involving pure parallel subsystems will be compared. Using LINGO solves the first ten examples of 100 pure parallel systems, the entire results are given in appendix F, and the comparison table is given.

Table 17 Comparisons of LINGO and GOZEBE

DIFFERENCES BETWEEN LINGO AND GOZEBE				
	RELIABILITY	COST	# of COMP	LINGO TIME
1	0.0059	476.87	-4	7.58
2	0.0088	304.32	-6	1.23
3	0.0081	-9.81	-1	9.88
4	0.0058	449.24	-4	2.37
5	0.0065	375.48	-4	4.20
6	0.0013	-6.12	-6	5.53
7	0.0079	119.34	1	0.53
8	0.0185	123.51	-2	30.72
9	0.0019	196.03	-2	0.87
10	0.0056	327.33	-7	1.85
mean=	0.0070	235.62	-3.50	6.48
std.dev.=	0.0047	177.28	2.51	9.06
CONFIDENCE INTERVALS of DIFF. and LINGO TIME				
alpha=	0.05			
	RELIABILITY	COST	# of COMP	TIME
UP	0.0106	368.20	-1.63	13.25
LOW	0.0035	103.03	-5.37	-0.30

The first column shows the differences between reliability values, the second column shows the differences between cost values, and the third column shows the differences between the number of components. The fourth column shows the solution wall clock time value if a Pentium III-1Ghz. CPU is used for LINGO solution. Confidence intervals show that there is a minor difference between GOZEBE heuristic and LINGO results based on reliability values. The upper bound is 0.0106, and the lower

bound is 0.0035. LINGO finds larger results everytime as expected but uses more money. The upper bound for using more money is $368.20, and the lower bound is $103.03. LINGO uses fewer components, only in example 7 it used 1 more component. The upper bound of using fewer components is 5.37, and the lower bound is 1.63.

The mean solution time for LINGO is 13.25 minutes, the solution time ranges from 0.53 minutes, which is 32 seconds, to 30.72 minutes, which is 30 minutes and 43 seconds. On the other hand, the new VBA software solves the same examples only a few seconds. The upper bound of solution time is 13.25 minutes and the lower bound is 0 minutes while it looks –0.3 in table 17 as a theoretical value. Therefore, it can be said that the new heuristic results are near optimum. Each example has 40 integer variables in its candidate sets totally. For larger problems, this solution times will increase if the capacity of LINGO enables for the number of integer variables. But, the capacity of LINGO can be exceeded by larger problems very quickly. Therefore, the new software is very important for design engineers.

V. Conclusions and Recommendations

5.1 Summary of This Research

This research aimed at reliability optimization, and covered the following issues. First, chapter one provided an introduction to the problem, research objectives, and methodology of this research, and gave an outline for this research. Chapter two reviewed engineering design, the importance of reliability in engineering design, reliability block diagrams, optimization tasks in reliability, reliability improvement methods, and other background materials. Chapter three sought a new heuristic to solve reliability optimization problem. Chapter four gave an explanation of how to use the software of the new heuristic and the results of randomly produced examples to show the performance of the new heuristic and to compare the new heuristic with marginal analysis heuristic.

Chapter five will discuss the weaknesses and strengths of the new heuristic, conclusions and recommendations, and future work sections.

5.2 Strengths and Weaknesses of the New Heuristic

The new heuristic basically takes its power from multiplication logic. In addition, the new heuristic selects the component with the largest marginal benefit ratio, instead of randomly selecting the components. These two processes are advantages of the new heuristic in finding the optimum solution. On the other hand, under limited budgets, some components, which have the maximum marginal benefit ratio, can have very a large cost, and this component consumes the budget very rapidly. In addition, misdirection of

multiplication logic mentioned in chapter three could produce different solutions from the optimum one.

Because the heuristics does not guarantee the optimum solution by definition, it is acceptable for a heuristic to find near optimum solutions in a reasonable time. Therefore, based on randomly produced examples, it can be said that the new heuristic works well as a greedy heuristic.

5.3 Conclusions and Recommendations

This research found a new heuristic, which gives good results. After examining 250 randomly produced examples, it can be concluded that this new heuristic finds larger results than marginal analysis heuristic based on success number, 69 to 51 in total. Moreover, it uses fewer components than marginal analysis does. Randomly produced examples showed that marginal analysis used more components than new heuristic in 44 examples, while the new heuristic used more components only one time because it started with a different initial design.

The software of this research can be used very easily. It finds the solution in only a few seconds. In addition to the new heuristic, the design engineer can use the software without considering which heuristic finds better result.

Design engineers, who know about VBA, can open the code and make changes fitting the special feature of the problem.

5.4 Future Research

This research examined the n-stage systems in which subsystems are in series structure to each other. Redundancy is made up on the initial system. This is called low-level redundancy. Future work can search for both complicated systems and high-level redundancy by using the same multiplication logic, and software can be written including these systems. These two systems will be explained with figures. An example of complicated systems is shown in figure 35.

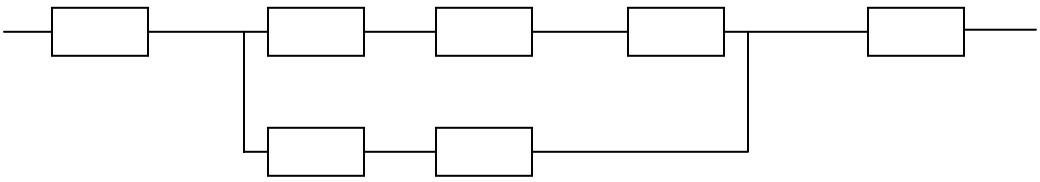


Figure 35 An Example of Complicated System

Each box represents different subsystems. In the middle of figure 35, there are two main branches, these branches can be reduced one reliability level, and the new heuristic can be implemented. What if the reduced middle part is selected at one step? Which branch has to be selected to be improved? The work showed that the branch, which has larger reliability value, has to be improved every time just like multiplication logic. Deeper research can be done, and new rules can be discovered. The high-level redundancy is shown in figure 36.

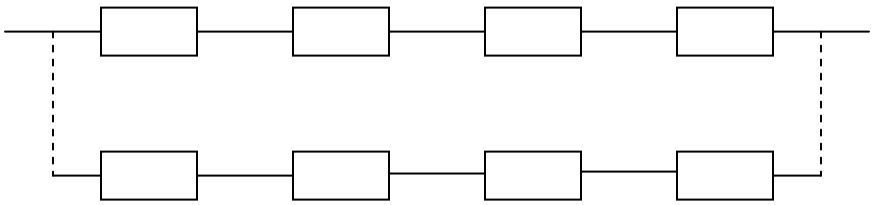


Figure 36 An Example of High-Level Redundancy

In figure 36 each box represents a different subsystem. The first row is the initial system design, and instead of component redundancy, the whole initial system can be doubled or more so that the overall system reliability improves. This process continues until there is no available budget remaining for high-level redundancy. If there is remaining budget, this can be used for low-level redundancy at the same time with high-level redundancy.

After finding solution steps of the new heuristic for either complicated systems or high-level redundancy, the steps can be coded so that design engineers can quickly find near optimum solutions.

Following the heuristic outlined in this thesis should allow for optimal system designs with maximum reliability under specific budget constraints.

Appendix A. VBA Codes of The New Software

Following subroutines are written in worksheet, named Menu.

```
Private Sub cmdBrowse_Click()
   'This macro lets the user to browse and open a data file
   Windows(ThisWorkbook.Name).Activate
   myName = ThisWorkbook.Name
   MYGLDIR = ActiveWorkbook.Path
   ChDir MYGLDIR
   mFilter = "All Files (*.*), *.*," & _
         "Excel Files (*.xls), *.xls"
   mFilterIndex = 2
   mTitle = "Select the Source File"
   FileAll = Application.GetOpenFilename(mFilter, mFilterIndex, mTitle)
   If FileAll = False Then
      MsgBox "No File Selected"
      Exit Sub
   End If
   txtFileName.Text = FileAll
   Workbooks.Open FileName:=FileAll
   Windows(myName).Activate
End Sub

Private Sub cmdExit_Click()
   'This macro saves and close the main menu file
   Range("a1").Select
   shMenu.txtFileName.Text = ""
   ActiveWorkbook.Save
   ActiveWorkbook.Close
End Sub

Private Sub cmdHeuristic_Click()
   'This macro first extracts the name of the data file and then run the heuristic
   myName = ThisWorkbook.Name
   FileSource = txtFileName.Text
   For SlashLoc = Len(FileSource) To 1 Step -1
      If Mid(FileSource, SlashLoc, 1) = "\" Then
         Exit For
      End If
   Next
   FileNameS = Mid(FileSource, SlashLoc + 1, Len(FileSource) - SlashLoc)
   Windows(FileNameS).Activate
   ReadProblemData
   FindBestRatio
```

```
    FindHighestReliability
    FindCheapestCost
    FindSolutionHG
    FindSolutionMA
    Sheets("Data").Activate
    ActiveSheet.Range("A1").Select
    ActiveWorkbook.Save
    Windows(myName).Activate
End Sub

Private Sub cmdSaveClose_Click()
    'This macro saves the data file and close it
    FileSource = txtFileName.Text
    myName = ThisWorkbook.Name
    For SlashLoc = Len(FileSource) To 1 Step -1
        If Mid(FileSource, SlashLoc, 1) = "\" Then
            Exit For
        End If
    Next
    FileNameS = Mid(FileSource, SlashLoc + 1, Len(FileSource) - SlashLoc)
    Windows(FileNameS).Activate
    Sheets("Data").Activate
    ActiveSheet.Range("A1").Select
    ActiveWorkbook.Save
    ActiveWorkbook.Close
    Windows(myName).Activate
    txtFileName.Text = ""
End Sub
```

Following subroutines are written workbook modules. There are six modules in total. First module, named DataDefinition, is for data definition.

```
Type COMP
    num As Integer
    rel As Double
    cost As Double
    ratio As Double
    subs As Integer
End Type

Type SUBSYS
    num As Integer
    k As Integer
    n As Integer
End Type
```

```
Type INITIAL_SOLUTION
   sub_list() As Integer
   comp_list() As Integer
   k_list() As Integer
   n_list() As Integer
   rel_list() As Double
   cost_list() As Double
   sys_rel As Double
   sys_cost As Double
   comp_cnt() As Integer
   comp_cnt_bk() As Integer
End Type

Global ncomp As Integer
Global nsubs As Integer
Global mybudget As Double
Global datasheet As String
Global initialsol As String
Global ratiosolsheet As String
Global relsolsheet As String
Global costsolsheet As String
Global mycomp() As COMP
Global mysubs() As SUBSYS
Global bestratio As INITIAL_SOLUTION
Global bestrel As INITIAL_SOLUTION
Global cheapest As INITIAL_SOLUTION
```

Following module, named DataRead, is for data reading.

```
Sub ReadProblemData()
   datasheet = "Data"
   initialsol = "Initial Solution"
   ratiosolsheet = "Solution-Ratio"
   relsolsheet = "Solution-Reliability"
   costsolsheet = "Solution-Cost"
   Sheets(datasheet).Activate
   'Read component data
   ncomp = 0
   rbeg = 3 'current subsystem data starts at row 3
   rend = rbeg 'end row is not known yet. We start it at rbeg and then increase it
   cbeg = 2 'column 2 or column B has the number of the susbsystem
   'Find the number of components and then dynamicallyallocate memory for the arrays
   Do While Cells(rend, cbeg) <> ""
      ncomp = ncomp + 1
```

```
    rend = rend + 1
  Loop
  ReDim mycomp(ncomp) As COMP
  'We sort the components by the subsystem they are assigned to (currently column E)
  Range("C" & rbeg - 1 & ":E" & rend - 1).Select
  Selection.Sort Key1:=Range("E" & rbeg), Order1:=xlAscending, Key2:=Range("C" &
rbeg) _
    , Order2:=xlAscending, Key3:=Range("D" & rbeg), Order3:=xlAscending,
Header:= _
    xlGuess, OrderCustom:=1, MatchCase:=False, Orientation:=xlTopToBottom
  Range("A1").Select
  For i = 0 To ncomp - 1
    mycomp(i).num = Cells(rbeg + i, cbeg)
    mycomp(i).rel = Cells(rbeg + i, cbeg + 1)
    mycomp(i).cost = Cells(rbeg + i, cbeg + 2)
    mycomp(i).subs = Cells(rbeg + i, cbeg + 3)
    mycomp(i).ratio = mycomp(i).rel / mycomp(i).cost
  Next i
  'Read subsystem data with k and n vectors
  nsubs = 0
  rbeg = 3
  rend = rbeg
  cbeg = 7
  Do While Cells(rend, cbeg) <> ""
    nsubs = nsubs + 1
    rend = rend + 1
  Loop
  ReDim mysubs(nsubs) As SUBSYS
  For i = 0 To nsubs - 1
    mysubs(i).num = Cells(rbeg + i, cbeg)
    mysubs(i).k = Cells(rbeg + i, cbeg + 1)
    mysubs(i).n = Cells(rbeg + i, cbeg + 2)
  Next i
  'Read budget
  mybudget = Range("budget")
  'Clean sheets
  Sheets(initialsol).Activate
  Range("B6:U1000").Select
  Selection.ClearContents
  Range("F4:G4").Select
  Selection.ClearContents
  Range("M4:N4").Select
  Selection.ClearContents
  Range("T4:U4").Select
  Selection.ClearContents
```

```
    Range("A1").Select
    Sheets(ratiosolsheet).Activate
    Range("B6:AD51000").Select
    Selection.ClearContents
    Range("H4:I4").Select
    Selection.ClearContents
    Range("Q4:R4").Select
    Selection.ClearContents
    Range("W4:X4").Select
    Selection.ClearContents
    Range("AC4:AD4").Select
    Selection.ClearContents
    Range("A1").Select
    Sheets(relsolsheet).Activate
    Range("B6:AD51000").Select
    Selection.ClearContents
    Range("H4:I4").Select
    Selection.ClearContents
    Range("Q4:R4").Select
    Selection.ClearContents
    Range("W4:X4").Select
    Selection.ClearContents
    Range("AC4:AD4").Select
    Selection.ClearContents
    Range("A1").Select
    Sheets(costsolsheet).Activate
    Range("B6:AD51000").Select
    Selection.ClearContents
    Range("H4:I4").Select
    Selection.ClearContents
    Range("Q4:R4").Select
    Selection.ClearContents
    Range("W4:X4").Select
    Selection.ClearContents
    Range("AC4:AD4").Select
    Selection.ClearContents
    Range("A1").Select
    Sheets(datasheet).Activate
End Sub
```

Following module, named FindInitialSolution, is for finding intial solution.

```
Sub FindBestRatio()
    ReDim bestratio.sub_list(nsubs) As Integer
    ReDim bestratio.comp_list(nsubs) As Integer
```

```
ReDim bestratio.k_list(nsubs) As Integer
ReDim bestratio.n_list(nsubs) As Integer
ReDim bestratio.rel_list(nsubs) As Double
ReDim bestratio.cost_list(nsubs) As Double
ReDim bestratio.comp_cnt(nsubs, ncomp) As Integer
ReDim bestratio.comp_cnt_bk(nsubs, ncomp) As Integer
Dim best_list() As Integer
ReDim best_list(nsubs) As Integer
Sheets(initialsol).Activate
For i = 0 To nsubs - 1
   curbestratio = -999
   curbestcompix = -1
   For j = 0 To ncomp - 1
      If mycomp(j).subs = mysubs(i).num And mycomp(j).ratio > curbestratio Then
         curbestratio = mycomp(j).ratio
         curbestcompix = j
      End If
      bestratio.comp_cnt(i, j) = 0
   Next j
   best_list(i) = curbestcompix
Next i
bestratio.sys_cost = 0
bestratio.sys_rel = 1
For i = 0 To nsubs - 1
   bestratio.sub_list(i) = mysubs(i).num
   bestratio.comp_list(i) = best_list(i)
   bestratio.k_list(i) = mysubs(i).k
   bestratio.n_list(i) = mysubs(i).n
   bestratio.rel_list(i) = INITIAL_SYSREL(mysubs(i).k, mysubs(i).n, best_list(i))
   bestratio.cost_list(i) = mysubs(i).n * mycomp(best_list(i)).cost
   bestratio.sys_cost = bestratio.sys_cost + bestratio.cost_list(i)
   bestratio.sys_rel = bestratio.sys_rel * bestratio.rel_list(i)
   bestratio.comp_cnt(i, best_list(i)) = mysubs(i).n
Next i
sys_result_row = 4
sys_result_col = 6
sys_data_row = 6
sys_data_col = 2
Sheets(initialsol).Cells(sys_result_row, sys_result_col) = bestratio.sys_rel
Sheets(initialsol).Cells(sys_result_row, sys_result_col + 1) = bestratio.sys_cost
sol_result_col = 8
sol_data_col = 2
Sheets(ratiosolsheet).Cells(sys_result_row, sol_result_col) = bestratio.sys_rel
Sheets(ratiosolsheet).Cells(sys_result_row, sol_result_col + 1) = bestratio.sys_cost
Sheets(ratiosolsheet).Cells(sys_result_row, sol_result_col + 9) = bestratio.sys_rel
```

```
   Sheets(ratiosolsheet).Cells(sys_result_row, sol_result_col + 10) = bestratio.sys_cost
   For i = 0 To nsubs - 1
   Sheets(initialsol).Cells(sys_data_row + i, sys_data_col) = bestratio.sub_list(i)
   Sheets(initialsol).Cells(sys_data_row + i, sys_data_col + 1) = bestratio.comp_list(i) +
1
   Sheets(initialsol).Cells(sys_data_row + i, sys_data_col + 2) = bestratio.k_list(i)
      Sheets(initialsol).Cells(sys_data_row + i, sys_data_col + 3) = bestratio.n_list(i)
      Sheets(initialsol).Cells(sys_data_row + i, sys_data_col + 4) = bestratio.rel_list(i)
      Sheets(initialsol).Cells(sys_data_row + i, sys_data_col + 5) = bestratio.cost_list(i)
      Sheets(ratiosolsheet).Cells(sys_data_row + i, sol_data_col) = "Initial"
      Sheets(ratiosolsheet).Cells(sys_data_row + i, sol_data_col + 1) = i + 1
      Sheets(ratiosolsheet).Cells(sys_data_row + i, sol_data_col + 2) =
bestratio.sub_list(i)
      Sheets(ratiosolsheet).Cells(sys_data_row + i, sol_data_col + 3) =
bestratio.comp_list(i) + 1
      Sheets(ratiosolsheet).Cells(sys_data_row + i, sol_data_col + 4) = bestratio.k_list(i)
      Sheets(ratiosolsheet).Cells(sys_data_row + i, sol_data_col + 5) = bestratio.n_list(i)
      Sheets(ratiosolsheet).Cells(sys_data_row + i, sol_data_col + 6) = bestratio.rel_list(i)
      Sheets(ratiosolsheet).Cells(sys_data_row + i, sol_data_col + 7) =
bestratio.cost_list(i)
      Sheets(ratiosolsheet).Cells(sys_data_row + i, sol_data_col + 9) = "Initial"
      Sheets(ratiosolsheet).Cells(sys_data_row + i, sol_data_col + 1 + 9) = i + 1
      Sheets(ratiosolsheet).Cells(sys_data_row + i, sol_data_col + 2 + 9) =
bestratio.sub_list(i)
      Sheets(ratiosolsheet).Cells(sys_data_row + i, sol_data_col + 3 + 9) =
bestratio.comp_list(i) + 1
      Sheets(ratiosolsheet).Cells(sys_data_row + i, sol_data_col + 4 + 9) =
bestratio.k_list(i)
      Sheets(ratiosolsheet).Cells(sys_data_row + i, sol_data_col + 5 + 9) =
bestratio.n_list(i)
      Sheets(ratiosolsheet).Cells(sys_data_row + i, sol_data_col + 6 + 9) =
bestratio.rel_list(i)
      Sheets(ratiosolsheet).Cells(sys_data_row + i, sol_data_col + 7 + 9) =
bestratio.cost_list(i)
   Next i
   For i = 0 To nsubs - 1
      For j = 0 To ncomp - 1
         bestratio.comp_cnt_bk(i, j) = bestratio.comp_cnt(i, j)
      Next j
   Next i
End Sub

Sub FindHighestReliability()
   ReDim bestrel.sub_list(nsubs) As Integer
   ReDim bestrel.comp_list(nsubs) As Integer
```

```
ReDim bestrel.k_list(nsubs) As Integer
ReDim bestrel.n_list(nsubs) As Integer
ReDim bestrel.rel_list(nsubs) As Double
ReDim bestrel.cost_list(nsubs) As Double
ReDim bestrel.comp_cnt(nsubs, ncomp) As Integer
ReDim bestrel.comp_cnt_bk(nsubs, ncomp) As Integer

Dim best_list() As Integer
ReDim best_list(nsubs) As Integer
Sheets(initialsol).Activate
For i = 0 To nsubs - 1
   curbestrel = -999
   curbestcompix = -1
   For j = 0 To ncomp - 1
      If mycomp(j).subs = mysubs(i).num And mycomp(j).rel > curbestrel Then
         curbestrel = mycomp(j).rel
         curbestcompix = j
      End If
      bestrel.comp_cnt(i, j) = 0
   Next j
   best_list(i) = curbestcompix
Next i
bestrel.sys_cost = 0
bestrel.sys_rel = 1
For i = 0 To nsubs - 1
   bestrel.sub_list(i) = mysubs(i).num
   bestrel.comp_list(i) = best_list(i)
   bestrel.k_list(i) = mysubs(i).k
   bestrel.n_list(i) = mysubs(i).n
   bestrel.rel_list(i) = INITIAL_SYSREL(mysubs(i).k, mysubs(i).n, best_list(i))
   bestrel.cost_list(i) = mysubs(i).n * mycomp(best_list(i)).cost
   bestrel.sys_cost = bestrel.sys_cost + bestrel.cost_list(i)
   bestrel.sys_rel = bestrel.sys_rel * bestrel.rel_list(i)
   bestrel.comp_cnt(i, best_list(i)) = mysubs(i).n
Next i
sys_result_row = 4
sys_result_col = 13
sys_data_row = 6
sys_data_col = 9
Sheets(initialsol).Cells(sys_result_row, sys_result_col) = bestrel.sys_rel
Sheets(initialsol).Cells(sys_result_row, sys_result_col + 1) = bestrel.sys_cost
sol_result_col = 8
sol_data_col = 2
Sheets(relsolsheet).Cells(sys_result_row, sol_result_col) = bestrel.sys_rel
Sheets(relsolsheet).Cells(sys_result_row, sol_result_col + 1) = bestrel.sys_cost
```

```
    Sheets(relsolsheet).Cells(sys_result_row, sol_result_col + 9) = bestrel.sys_rel
    Sheets(relsolsheet).Cells(sys_result_row, sol_result_col + 10) = bestrel.sys_cost
    For i = 0 To nsubs - 1
        Sheets(initialsol).Cells(sys_data_row + i, sys_data_col) = bestrel.sub_list(i)
        Sheets(initialsol).Cells(sys_data_row + i, sys_data_col + 1) = bestrel.comp_list(i)
+1
        Sheets(initialsol).Cells(sys_data_row + i, sys_data_col + 2) = bestrel.k_list(i)
        Sheets(initialsol).Cells(sys_data_row + i, sys_data_col + 3) = bestrel.n_list(i)
        Sheets(initialsol).Cells(sys_data_row + i, sys_data_col + 4) = bestrel.rel_list(i)
        Sheets(initialsol).Cells(sys_data_row + i, sys_data_col + 5) = bestrel.cost_list(i)
        Sheets(relsolsheet).Cells(sys_data_row + i, sol_data_col) = "Initial"
        Sheets(relsolsheet).Cells(sys_data_row + i, sol_data_col + 1) = i + 1
        Sheets(relsolsheet).Cells(sys_data_row + i, sol_data_col + 2) = bestrel.sub_list(i)
        Sheets(relsolsheet).Cells(sys_data_row + i, sol_data_col + 3) = bestrel.comp_list(i)
+ 1
        Sheets(relsolsheet).Cells(sys_data_row + i, sol_data_col + 4) = bestrel.k_list(i)
        Sheets(relsolsheet).Cells(sys_data_row + i, sol_data_col + 5) = bestrel.n_list(i)
        Sheets(relsolsheet).Cells(sys_data_row + i, sol_data_col + 6) = bestrel.rel_list(i)
        Sheets(relsolsheet).Cells(sys_data_row + i, sol_data_col + 7) = bestrel.cost_list(i)
        Sheets(relsolsheet).Cells(sys_data_row + i, sol_data_col + 9) = "Initial"
        Sheets(relsolsheet).Cells(sys_data_row + i, sol_data_col + 1 + 9) = i + 1
        Sheets(relsolsheet).Cells(sys_data_row + i, sol_data_col + 2 + 9) =
bestrel.sub_list(i)
        Sheets(relsolsheet).Cells(sys_data_row + i, sol_data_col + 3 + 9) =
bestrel.comp_list(i) + 1
        Sheets(relsolsheet).Cells(sys_data_row + i, sol_data_col + 4 + 9) = bestrel.k_list(i)
        Sheets(relsolsheet).Cells(sys_data_row + i, sol_data_col + 5 + 9) = bestrel.n_list(i)
        Sheets(relsolsheet).Cells(sys_data_row + i, sol_data_col + 6 + 9) = bestrel.rel_list(i)
        Sheets(relsolsheet).Cells(sys_data_row + i, sol_data_col + 7 + 9) =
bestrel.cost_list(i)
    Next i
    For i = 0 To nsubs - 1
        For j = 0 To ncomp - 1
            bestrel.comp_cnt_bk(i, j) = bestrel.comp_cnt(i, j)
        Next j
    Next i
End Sub

Sub FindCheapestCost()
    ReDim cheapest.sub_list(nsubs) As Integer
    ReDim cheapest.comp_list(nsubs) As Integer
    ReDim cheapest.k_list(nsubs) As Integer
    ReDim cheapest.n_list(nsubs) As Integer
    ReDim cheapest.rel_list(nsubs) As Double
    ReDim cheapest.cost_list(nsubs) As Double
```

```
ReDim cheapest.comp_cnt(nsubs, ncomp) As Integer
ReDim cheapest.comp_cnt_bk(nsubs, ncomp) As Integer
Dim best_list() As Integer
ReDim best_list(nsubs) As Integer
Sheets(initialsol).Activate
For i = 0 To nsubs - 1
   curcheapest = 1000000
   curbestcompix = -1

   For j = 0 To ncomp - 1
      If mycomp(j).subs = mysubs(i).num And mycomp(j).cost < curcheapest Then
         curcheapest = mycomp(j).rel
         curbestcompix = j
      End If
      cheapest.comp_cnt(i, j) = 0
   Next j
   best_list(i) = curbestcompix
Next i
cheapest.sys_cost = 0
cheapest.sys_rel = 1
For i = 0 To nsubs - 1
   cheapest.sub_list(i) = mysubs(i).num
   cheapest.comp_list(i) = best_list(i)
   cheapest.k_list(i) = mysubs(i).k
   cheapest.n_list(i) = mysubs(i).n
   cheapest.rel_list(i) = INITIAL_SYSREL(mysubs(i).k, mysubs(i).n, best_list(i))
   cheapest.cost_list(i) = mysubs(i).n * mycomp(best_list(i)).cost
   cheapest.sys_cost = cheapest.sys_cost + cheapest.cost_list(i)
   cheapest.sys_rel = cheapest.sys_rel * cheapest.rel_list(i)
   cheapest.comp_cnt(i, best_list(i)) = mysubs(i).n
Next i
sys_result_row = 4
sys_result_col = 20
sys_data_row = 6
sys_data_col = 16
Sheets(initialsol).Cells(sys_result_row, sys_result_col) = cheapest.sys_rel
Sheets(initialsol).Cells(sys_result_row, sys_result_col + 1) = cheapest.sys_cost
sol_result_col = 8
sol_data_col = 2
Sheets(costsolsheet).Cells(sys_result_row, sol_result_col) = cheapest.sys_rel
Sheets(costsolsheet).Cells(sys_result_row, sol_result_col + 1) = cheapest.sys_cost
Sheets(costsolsheet).Cells(sys_result_row, sol_result_col + 9) = cheapest.sys_rel
Sheets(costsolsheet).Cells(sys_result_row, sol_result_col + 10) = cheapest.sys_cost
For i = 0 To nsubs - 1
   Sheets(initialsol).Cells(sys_data_row + i, sys_data_col) = cheapest.sub_list(i)
```

```
        Sheets(initialsol).Cells(sys_data_row + i, sys_data_col + 1) = cheapest.comp_list(i)
+ 1
        Sheets(initialsol).Cells(sys_data_row + i, sys_data_col + 2) = cheapest.k_list(i)
        Sheets(initialsol).Cells(sys_data_row + i, sys_data_col + 3) = cheapest.n_list(i)
        Sheets(initialsol).Cells(sys_data_row + i, sys_data_col + 4) = cheapest.rel_list(i)
        Sheets(initialsol).Cells(sys_data_row + i, sys_data_col + 5) = cheapest.cost_list(i)
        Sheets(costsolsheet).Cells(sys_data_row + i, sol_data_col) = "Initial"
        Sheets(costsolsheet).Cells(sys_data_row + i, sol_data_col + 1) = i + 1
        Sheets(costsolsheet).Cells(sys_data_row + i, sol_data_col + 2) = cheapest.sub_list(i)
        Sheets(costsolsheet).Cells(sys_data_row + i, sol_data_col + 3) =
cheapest.comp_list(i) + 1
        Sheets(costsolsheet).Cells(sys_data_row + i, sol_data_col + 4) = cheapest.k_list(i)
        Sheets(costsolsheet).Cells(sys_data_row + i, sol_data_col + 5) = cheapest.n_list(i)
        Sheets(costsolsheet).Cells(sys_data_row + i, sol_data_col + 6) = cheapest.rel_list(i)
        Sheets(costsolsheet).Cells(sys_data_row + i, sol_data_col + 7) =
cheapest.cost_list(i)
        Sheets(costsolsheet).Cells(sys_data_row + i, sol_data_col + 9) = "Initial"
        Sheets(costsolsheet).Cells(sys_data_row + i, sol_data_col + 1 + 9) = i + 1
        Sheets(costsolsheet).Cells(sys_data_row + i, sol_data_col + 2 + 9) =
cheapest.sub_list(i)
        Sheets(costsolsheet).Cells(sys_data_row + i, sol_data_col + 3 + 9) =
cheapest.comp_list(i) + 1
        Sheets(costsolsheet).Cells(sys_data_row + i, sol_data_col + 4 + 9) =
cheapest.k_list(i)
        Sheets(costsolsheet).Cells(sys_data_row + i, sol_data_col + 5 + 9) =
cheapest.n_list(i)
        Sheets(costsolsheet).Cells(sys_data_row + i, sol_data_col + 6 + 9) =
cheapest.rel_list(i)
        Sheets(costsolsheet).Cells(sys_data_row + i, sol_data_col + 7 + 9) =
cheapest.cost_list(i)
    Next i
    For i = 0 To nsubs - 1
        For j = 0 To ncomp - 1
            cheapest.comp_cnt_bk(i, j) = cheapest.comp_cnt(i, j)
        Next j
    Next i
End Sub
```

Following module, named HeuristicHG, is used for GOZEBE heuristic.

```
Sub FindSolutionHG()

    HG_Solution_For bestratio, ratiosolsheet
    HG_Solution_For bestrel, relsolsheet
    HG_Solution_For cheapest, costsolsheet
```

```
End Sub

Sub HG_Solution_For(ByRef xinitial As INITIAL_SOLUTION, ByVal shsol As String)
   'Current system variables
   Dim csno() As Integer 'Stores the index of the subsystem i
   Dim csr1() As Double 'Stores the initial reliability of subsystem i
   Dim csr2() As Double 'Stores the current reliability of the subsystem as new
components are added
   Dim csc1() As Double 'Stores the initial cost of the subsystem
   Dim csc2() As Double 'Stores the current cost of the subsystem as new components are
added
   Dim csc3() As Double 'Keep the cost of subsystem for previous iteration
   Dim ncix() As Integer 'Stores the index of component with the best ratio
   Dim cskk() As Integer 'Stores k number for the subsystem
   Dim csnn() As Integer 'Stores n number for the initial subsystem
   Dim csnn2() As Integer 'Stores n number for the current subsystem
   Dim csii() As Integer 'Stores the initial component id for the subsystem
   Dim check() As Integer  'Stores 0/1 indicators to know if a subsytem is checked
                  'for improvement. First, the lowest reliability subsystem is
                  'selected and componets are added. Then, second lowest, third lowest,
etc.
                  'until all are checked.  When all are cheked for best component to add,
we
                  'return to a clean sheet again and start with second best components,
third, etc and so on
   ReDim csno(nsubs) As Integer
   ReDim csr1(nsubs) As Double
   ReDim csr2(nsubs) As Double
   ReDim csc1(nsubs) As Double
   ReDim csc2(nsubs) As Double
   ReDim csc3(nsubs) As Double
   ReDim ncix(nsubs) As Integer
   ReDim cskk(nsubs) As Integer
   ReDim csnn(nsubs) As Integer
   ReDim csnn2(nsubs) As Integer
   ReDim csii(nsubs) As Integer
   ReDim check(nsubs) As Integer
   'New components variables
   Dim ncno() As Integer 'Stores the index of the new component i
   Dim ncs() As Integer 'Stores the subsystem index for each new component
   Dim ncr() As Double 'Stores the reliability of component i
   Dim ncc() As Double 'Stores the cost of component i
   Dim nct() As Double 'Stores the reliability/cost ratio for each component i
   Dim compcheck() As Integer  'This is an indicator showing if a component is available
to use
```

```
                    'This will let us use the second best, third best, etc. based on
                    'the budget constraint
ReDim ncno(ncomp) As Integer
ReDim ncs(ncomp) As Integer
ReDim ncr(ncomp) As Double
ReDim ncc(ncomp) As Double
ReDim nct(ncomp) As Double
ReDim compcheck(ncomp) As Integer
Dim extbudget As Double 'Maximum extra budget to spend

'Now we know the exact number of subsystems.  We read the subsystem data
For i = 0 To nsubs - 1
   csno(i) = xinitial.sub_list(i)
   csr1(i) = xinitial.rel_list(i)
   csc1(i) = xinitial.cost_list(i)
   cskk(i) = xinitial.k_list(i)
   csnn(i) = xinitial.n_list(i)
   csnn2(i) = csnn(i)
   csii(i) = xinitial.comp_list(i)
   csr2(i) = csr1(i)
   csc2(i) = csc1(i)
Next i
'We read the subsystem data and calculate the relibility/cost ratio
For i = 0 To ncomp - 1
   ncno(i) = mycomp(i).num
   ncr(i) = mycomp(i).rel
   ncc(i) = mycomp(i).cost
   ncs(i) = mycomp(i).subs
   nct(i) = ncr(i) / ncc(i)
Next i
'we find the extra budget
extbudget = mybudget - xinitial.sys_cost
For i = 0 To nsubs - 1
   For j = 0 To ncomp - 1
      xinitial.comp_cnt(i, j) = xinitial.comp_cnt_bk(i, j)
   Next j
Next i
totrel = 1  'this is the total reliability of the system.  Initially, it is
         'assigned to 1 since 1 has no affect on the multiplication
totcompcheck = 0    'this is a cound for the components used.  At one point,
                'after all components (best, second best, third best, etc.
                'are checked in order to add, the program should exit
'Since I started using randomly generated component reliabilities,
'we should check if there is any with 0 reliability.  It should be checked
'as "not available" at the beginning
```

```
For i = 0 To ncomp - 1
   If ncr(i) <> 0 Then
      compcheck(i) = 0
   Else
      compcheck(i) = 1
      totcompcheck = totcompcheck + 1
   End If
Next i
totcost = 0 'This is the variable to store the total extra cost of the
          'components to be added

finalsolrow = 6 + nsubs 'this is the beginning row in the solution sheet
rrbeg = 4 'this is the row that stores the cumulative numbers in the solution sheet
cur_iter_num = 0
num_comp_added = -1
Do While totcompcheck < ncomp   'this check if all components are checked if
                                'they could be added to the system
   cur_iter_num = cur_iter_num + 1
   If num_comp_added = -1 Or num_comp_added > 0 Then
      finalsolrow = finalsolrow + 1
   End If
   num_comp_added = 0
   'Now find the best component for each subsystem.  This finds the lowest
   'available one.  If there is a tie, it selects the one with the best
   'reliability
   For i = 0 To nsubs - 1
      ncix(i) = -1
      maxratio = -999
      If cskk(i) > 1 Then
         If compcheck(csii(i)) = 0 Then
            ncix(i) = csii(i)
            maxratio = nct(csii(i))
         End If
         For j = 0 To ncomp - 1
            If ncs(j) = i + 1 And j <> csii(i) Then
               compcheck(j) = 1
               totcompcheck = totcompcheck + 1
            End If
         Next j
      Else
         For j = 0 To ncomp - 1
            If ncs(j) = i + 1 Then
               If nct(j) > maxratio And compcheck(j) = 0 Then
                  ncix(i) = j
                  maxratio = nct(j)
```

```
                Else
                    If nct(j) = maxratio And compcheck(j) = 0 And ncix(i) <> -1 Then
                        If ncr(j) > ncr(ncix(i)) Then
                            ncix(i) = j
                            maxratio = nct(j)
                        End If
                    End If
                End If
            End If
        Next j
    End If
Next i

'At the beginning all subsystems are available
For i = 0 To nsubs - 1
    check(i) = 0
    csc3(i) = csc2(i)
Next i
loopexit = 0 'this is a variable to exit the loop when necessary
'this loop continues as long as the extra cost of the added components
'is under the budget and loopexit is zero. AT one point, there may not be
'any component available to add even though the extyra cost is still under
'the budget.  For this reason, we need to use loopexit variable.
cur_step_num = 0
Do While totcost <= extbudget And loopexit = 0
    'cur_step_num = cur_step_num + 1
    addcost = 0 'this is the cost of a candidate component to add
    minrel = 999 'high number to find the subsystem to improve
    minsubix = -1   'index of subsystem to improve.  It is assigned a
                'negative number so that we will know if there is any
                'available subsystem to improve
  'Now find the subsystem with lowest reliability. We have to find the
  'lowest among the available ones at each iteration since some of them may
  'have been already visited.  If there is a tie, the one with the
  'best component to add is selected
   For i = 0 To nsubs - 1
      If minrel > csr2(i) And check(i) = 0 Then
         minsubix = i
         minrel = csr2(i)
      Else
         If minrel = csr2(i) And check(i) = 0 Then
            If minsubix <> -1 And ncix(i) <> -1 And ncix(minsubix) <> -1 Then
               If compcheck(ncix(i)) = 0 And compcheck(ncix(minsubix)) = 0 And
totcost + ncc(ncix(i)) <= extbudget Then
                  If nct(ncix(i)) > nct(ncix(minsubix)) Then
```

```
                        minsubix = i
                        minrel = csr2(i)
                     Else
                        If nct(ncix(i)) = nct(ncix(minsubix)) Then
                           If ncr(ncix(i)) > ncr(ncix(minsubix)) Then
                              minsubix = i
                              minrel = csr2(i)
                           End If
                        End If
                     End If
                  End If
               End If
            End If
         Next i
         'If there is a subsystem available to improve, then we proceed.
         'Otherwise, we exit the loop and start with the second best,
         'third best, etc. set of components
         If minsubix <> -1 Then
            'If the subsystem is available, then we need to check the
            'best component to be added to that subsystem. If all components
            'for that subsystem are evaluated before, then we proceed, otherwise
            'we need to go to another subsystem and make this one unavailable
            'by assigning check(minsubix)=1
            If ncix(minsubix) <> -1 Then
               'First, we calculate the anticipated reliability of the subsystem
               'when this component is added.  We also find the extra cost
               'will add
               If cskk(minsubix) = 1 Then
                  newsubrel = 1 - (1 - csr2(minsubix)) * (1 - ncr(ncix(minsubix)))
               Else
                  newsubrel = INITIAL_SYSREL(cskk(minsubix), csnn2(minsubix) + 1,
ncix(minsubix))
               End If
               newsubcst = csc2(minsubix) + ncc(ncix(minsubix))
               'If the total extra cost plus the cost of the component being
               'added is under the budget, we will add that component.
               'Otherwise, that component will become unavailable for following
               'iterations and then we will continue
               If totcost + ncc(ncix(minsubix)) <= extbudget Then
                  cur_step_num = cur_step_num + 1
                  csr2(minsubix) = newsubrel 'update the subsystem reliability
                  csc2(minsubix) = newsubcst 'update the subsystem cost
                  If cskk(minsubix) > 1 Then
                     csnn2(minsubix) = csnn2(minsubix) + 1
```

```
                End If
                xinitial.comp_cnt(minsubix, ncix(minsubix)) =
xinitial.comp_cnt(minsubix, ncix(minsubix)) + 1
                addcost = ncc(ncix(minsubix)) 'Find extra cost beinf added
                totcost = totcost + addcost 'Calculate the total extra cost
                'Now write iteration number, step number and other data
                Sheets(shsol).Cells(rrbeg, 8) = ARRAY_SYSREL(csr2())
                Sheets(shsol).Cells(rrbeg, 9) = ARRAY_SYSCOST(csc2())
                Sheets(shsol).Cells(finalsolrow, 2) = cur_iter_num
                Sheets(shsol).Cells(finalsolrow, 3) = cur_step_num
                Sheets(shsol).Cells(finalsolrow, 4) = minsubix + 1
                Sheets(shsol).Cells(finalsolrow, 5) = ncix(minsubix) + 1
                Sheets(shsol).Cells(finalsolrow, 6) = "-"
                Sheets(shsol).Cells(finalsolrow, 7) = "-"
                Sheets(shsol).Cells(finalsolrow, 8) = ncr(ncix(minsubix))
                Sheets(shsol).Cells(finalsolrow, 9) = ncc(ncix(minsubix))
                finalsolrow = finalsolrow + 1
                num_comp_added = num_comp_added + 1
             Else
                check(minsubix) = 1 'Make the subsystem unavailable, e.g. visited for this
iteration
                compcheck(ncix(minsubix)) = 1 'Make the component unavailable
                totcompcheck = totcompcheck + 1 'Increase the total number of
unavailable components by 1
             End If
          Else
             check(minsubix) = 1 'Make the subsytem unavailable, e.g. visited
                         'for this iteration
          End If
       Else
          loopexit = 1 'Exit the loop
       End If
    Loop
    If cur_iter_num = 1 And totcompcheck >= ncomp Then
       finalsolrow = finalsolrow + 1
    End If
 Loop
 'Write final reliability ands costs of subsystems
 For i = 0 To nsubs - 1
    Sheets(shsol).Cells(finalsolrow + i, 2) = "Final"
    Sheets(shsol).Cells(finalsolrow + i, 3) = i + 1
    Sheets(shsol).Cells(finalsolrow + i, 4) = i + 1
    Sheets(shsol).Cells(finalsolrow + i, 5) = "-"
    Sheets(shsol).Cells(finalsolrow + i, 6) = "-"
    Sheets(shsol).Cells(finalsolrow + i, 7) = "-"
```

```
      Sheets(shsol).Cells(finalsolrow + i, 8) = csr2(i)
      Sheets(shsol).Cells(finalsolrow + i, 9) = csc2(i)
   Next i
   'Write the solution summary
   sumrowbeg = 6
   jpositive = 0
   For i = 0 To nsubs - 1
      Sheets(shsol).Cells(rrbeg, 23) = Sheets(shsol).Cells(rrbeg, 8)
      Sheets(shsol).Cells(rrbeg, 24) = Sheets(shsol).Cells(rrbeg, 9)
      For j = 0 To ncomp - 1
         If xinitial.comp_cnt(i, j) > 0 Then
            Sheets(shsol).Cells(sumrowbeg + jpositive, 20) = i + 1
            Sheets(shsol).Cells(sumrowbeg + jpositive, 21) = j + 1
            Sheets(shsol).Cells(sumrowbeg + jpositive, 22) = xinitial.comp_cnt(i, j)
            Sheets(shsol).Cells(sumrowbeg + jpositive, 23) = mycomp(j).rel
            Sheets(shsol).Cells(sumrowbeg + jpositive, 24) = mycomp(j).cost
            jpositive = jpositive + 1
         End If
      Next j
   Next i
End Sub
```

Following module, named HeuristicMA, is used for marginal analysis heuristic.

```
Sub FindSolutionMA()
   MA_Solution_For bestratio, ratiosolsheet
   MA_Solution_For bestrel, relsolsheet
   MA_Solution_For cheapest, costsolsheet
End Sub

Sub MA_Solution_For(ByRef xinitial As INITIAL_SOLUTION, ByVal shsol As String)
   'Current system variables
   Dim csno() As Integer 'Stores the index of the subsystem i
   Dim csr1() As Double 'Stores the initial reliability of subsystem i
   Dim csr2() As Double 'Stores the current reliability of the subsystem as new
components are added
   Dim csc1() As Double 'Stores the initial cost of the subsystem
   Dim csc2() As Double 'Stores the current cost of the subsystem as new components are
added
   Dim csc3() As Double 'Keep the cost of subsystem for previous iteration
   Dim ncix() As Integer 'Stores the index of component with the best ratio
   Dim cskk() As Integer 'Stores k number for the subsystem
   Dim csnn() As Integer 'Stores n number for the initial subsystem
   Dim csnn2() As Integer 'Stores n number for the current subsystem
```

```
   Dim csii() As Integer 'Stores the initial component id for the subsystem
   Dim check() As Integer  'Stores 0/1 indicators to know if a subsytem is checked
                           'for improvement. First, the lowest reliability subsystem is
                           'selected and componets are added. Then, second lowest, third lowest,
etc.
                           'until all are checked.  When all are cheked for best component to add,
we
                           'return to a clean sheet again and start with second best components,
third, etc and so on
   ReDim csno(nsubs) As Integer
   ReDim csr1(nsubs) As Double
   ReDim csr2(nsubs) As Double
   ReDim csc1(nsubs) As Double
   ReDim csc2(nsubs) As Double
   ReDim csc3(nsubs) As Double
   ReDim ncix(nsubs) As Integer
   ReDim cskk(nsubs) As Integer
   ReDim csnn(nsubs) As Integer
   ReDim csnn2(nsubs) As Integer
   ReDim csii(nsubs) As Integer
   ReDim check(nsubs) As Integer
   'New components variables
   Dim ncno() As Integer 'Stores the index of the new component i
   Dim ncs() As Integer 'Stores the subsystem index for each new component
   Dim ncr() As Double 'Stores the reliability of component i
   Dim ncc() As Double 'Stores the cost of component i
   Dim nct() As Double 'Stores the reliability/cost ratio for each component i
   Dim compcheck() As Integer  'This is an indicator showing if a component is available
to use
                           'This will let us use the second best, third best, etc. based on
                           'the budget constraint
   ReDim ncno(ncomp) As Integer
   ReDim ncs(ncomp) As Integer
   ReDim ncr(ncomp) As Double
   ReDim ncc(ncomp) As Double
   ReDim nct(ncomp) As Double
   ReDim compcheck(ncomp) As Integer
   Dim extbudget As Double 'Maximum extra budget to spend
   'Now we know the exact number of subsystems.  We read the subsystem data
   For i = 0 To nsubs - 1
      csno(i) = xinitial.sub_list(i)
      csr1(i) = xinitial.rel_list(i)
      csc1(i) = xinitial.cost_list(i)
      cskk(i) = xinitial.k_list(i)
      csnn(i) = xinitial.n_list(i)
```

```
    csnn2(i) = csnn(i)
    csii(i) = xinitial.comp_list(i)
    csr2(i) = csr1(i)
    csc2(i) = csc1(i)
Next i
'We read the subsystem data and calculate the relibility/cost ratio
For i = 0 To ncomp - 1
    ncno(i) = mycomp(i).num
    ncr(i) = mycomp(i).rel
    ncc(i) = mycomp(i).cost
    ncs(i) = mycomp(i).subs
    nct(i) = ncr(i) / ncc(i)
Next i

'we find the extra budget
extbudget = mybudget - xinitial.sys_cost
For i = 0 To nsubs - 1
    For j = 0 To ncomp - 1
        xinitial.comp_cnt(i, j) = xinitial.comp_cnt_bk(i, j)
    Next j
Next i
totrel = 1  'this is the total reliability of the system.  Initially, it is
          'assigned to 1 since 1 has no affect on the multiplication
totcompcheck = 0    'this is a cound for the components used.  At one point,
                    'after all components (best, second best, third best, etc.
                    'are checked in order to add, the program should exit
'Since I started using randomly generated component reliabilities,
'we should check if there is any with 0 reliability.  It should be checked
'as "not available" at the beginning
For i = 0 To ncomp - 1
    If ncr(i) <> 0 Then
        compcheck(i) = 0
    Else
        compcheck(i) = 1
        totcompcheck = totcompcheck + 1
    End If
Next i
totcost = 0 'This is the variable to store the total extra cost of the
          'components to be added
finalsolrow = 6 + nsubs 'this is the beginning row in the solution sheet
rrbeg = 4 'this is the row that stores the cumulative numbers in the solution sheet
cur_iter_num = 0
num_comp_added = -1
Do While totcompcheck < ncomp   'this check if all components are checked if
                                'they could be added to the system
```

```
cur_iter_num = cur_iter_num + 1
If num_comp_added = -1 Or num_comp_added > 0 Then
   finalsolrow = finalsolrow + 1
End If
num_comp_added = 0
'Now find the best component for each subsystem.  This adds the component
'to the subsystem, find the increase in reliability and then divide it
'by the cost of the component.
For i = 0 To nsubs - 1
   ncix(i) = -1
   maxratio = -999
   If cskk(i) > 1 Then
      If compcheck(csii(i)) = 0 Then
         ncix(i) = csii(i)
         maxratio = nct(csii(i))
      End If
      For j = 0 To ncomp - 1
         If ncs(j) = i + 1 And j <> csii(i) Then
            compcheck(j) = 1
            totcompcheck = totcompcheck + 1
         End If
      Next j
   Else
      For j = 0 To ncomp - 1
         If ncs(j) = i + 1 Then
            If compcheck(j) = 0 Then
               newrel = 1 - (1 - csr2(i)) * (1 - ncr(j))
               relinc = Log(newrel) - Log(csr2(i))
               incratio = relinc / ncc(j)
               If incratio > maxratio Then
                  ncix(i) = j
                  maxratio = incratio
               Else
                  If incratio = maxratio And ncix(i) <> -1 Then
                     If ncr(j) > ncr(ncix(i)) Then
                        ncix(i) = j
                        maxratio = incratio
                     End If
                  End If
               End If
            End If
         End If
      Next j
   End If
Next i
```

```
'At the beginning all subsystems are available
For i = 0 To nsubs - 1
   check(i) = 0
   csc3(i) = csc2(i)
Next i
loopexit = 0 'this is a variable to exit the loop when necessary
'this loop continues as long as the extra cost of the added components
'is under the budget and loopexit is zero. AT one point, there may not be
'any component available to add even though the extyra cost is still under
'the budget.  For this reason, we need to use loopexit variable.
cur_step_num = 0
Do While totcost <= extbudget And loopexit = 0
   'cur_step_num = cur_step_num + 1

   addcost = 0 'this is the cost of a candidate component to add
   minrel = -999 'low number to find the subsystem to improve
   minsubix = -1   'index of subsystem to improve.  It is assigned a
               'negative number so that we will know if there is any
               'available subsystem to improve
   lowindex = -1
  'Now find the subsystem with highest rate of increase.
   For i = 0 To nsubs - 1
      If ncix(i) <> -1 Then
         newrel = 1 - (1 - csr2(i)) * (1 - ncr(ncix(i)))
         relinc = Log(newrel) - Log(csr2(i))
         incratio = relinc / ncc(ncix(i))
         If incratio > minrel And check(i) = 0 Then
            minsubix = i
            minrel = incratio
            lowindex = i
         Else
            If incratio = minrel And check(i) = 0 Then
               If i < lowindex Then
                  minsubix = i
                  minrel = incratio
                  lowindex = i
               End If
            End If
         End If
      End If
   Next i
   'If there is a subsystem available to improve, then we proceed.
   'Otherwise, we exit the loop and start with the second best,
   'third best, etc. set of components
   If minsubix <> -1 Then
```

```
            'If the subsystem is available, then we need to check the
            'best component to be added to that subsystem. If all components
            'for that subsystem are evaluated before, then we proceed, otherwise
            'we need to go to another subsystem and make this one unavailable
            'by assigning check(minsubix)=1
            If ncix(minsubix) <> -1 Then
               'First, we calculate the anticipated reliability of the subsystem
               'when this component is added.  We also find the extra cost
               'will add
               If cskk(minsubix) = 1 Then
                  newsubrel = 1 - (1 - csr2(minsubix)) * (1 - ncr(ncix(minsubix)))
               Else
                  newsubrel = INITIAL_SYSREL(cskk(minsubix), csnn2(minsubix) + 1,
ncix(minsubix))
               End If
               newsubcst = csc2(minsubix) + ncc(ncix(minsubix))
               'If the total extra cost plus the cost of the component being
               'added is under the budget, we will add that component.
               'Otherwise, that component will become unavailable for following
               'iterations and then we will continue
               If totcost + ncc(ncix(minsubix)) <= extbudget Then
                  cur_step_num = cur_step_num + 1
                  csr2(minsubix) = newsubrel 'update the subsystem reliability
                  csc2(minsubix) = newsubcst 'update the subsystem cost
                  If cskk(minsubix) > 1 Then
                     csnn2(minsubix) = csnn2(minsubix) + 1
                  End If
                  xinitial.comp_cnt(minsubix, ncix(minsubix)) =
xinitial.comp_cnt(minsubix, ncix(minsubix)) + 1
                  addcost = ncc(ncix(minsubix)) 'Find extra cost beinf added
                  totcost = totcost + addcost 'Calculate the total extra cost
                  'Now write iteration number, step number and other data
                  Sheets(shsol).Cells(rrbeg, 17) = ARRAY_SYSREL(csr2())
                  Sheets(shsol).Cells(rrbeg, 18) = ARRAY_SYSCOST(csc2())
                  Sheets(shsol).Cells(finalsolrow, 11) = cur_iter_num
                  Sheets(shsol).Cells(finalsolrow, 12) = cur_step_num
                  Sheets(shsol).Cells(finalsolrow, 13) = minsubix + 1
                  Sheets(shsol).Cells(finalsolrow, 14) = ncix(minsubix) + 1
                  Sheets(shsol).Cells(finalsolrow, 15) = "-"
                  Sheets(shsol).Cells(finalsolrow, 16) = "-"
                  Sheets(shsol).Cells(finalsolrow, 17) = ncr(ncix(minsubix))
                  Sheets(shsol).Cells(finalsolrow, 18) = ncc(ncix(minsubix))
                  finalsolrow = finalsolrow + 1
                  num_comp_added = num_comp_added + 1
               Else
```

```
                check(minsubix) = 1 'Make the subsystem unavailable, e.g. visited for this
iteration
                compcheck(ncix(minsubix)) = 1 'Make the component unavailable
                totcompcheck = totcompcheck + 1 'Increase the total number of
unavailable components by 1
              End If
            Else
              check(minsubix) = 1 'Make the subsytem unavailable, e.g. visited
                    'for this iteration
            End If
          Else
            loopexit = 1 'Exit the loop
          End If
        Loop

        If cur_iter_num = 1 And totcompcheck >= ncomp Then
          finalsolrow = finalsolrow + 1
        End If
      Loop
      'Write final reliability ands costs of subsystems
      For i = 0 To nsubs - 1
        Sheets(shsol).Cells(finalsolrow + i, 11) = "Final"
        Sheets(shsol).Cells(finalsolrow + i, 12) = i + 1
        Sheets(shsol).Cells(finalsolrow + i, 13) = i + 1
        Sheets(shsol).Cells(finalsolrow + i, 14) = "-"
        Sheets(shsol).Cells(finalsolrow + i, 15) = "-"
        Sheets(shsol).Cells(finalsolrow + i, 16) = "-"
        Sheets(shsol).Cells(finalsolrow + i, 17) = csr2(i)
        Sheets(shsol).Cells(finalsolrow + i, 18) = csc2(i)
      Next i
      'Write the solution summary
      sumrowbeg = 6
      jpositive = 0
      For i = 0 To nsubs - 1
        Sheets(shsol).Cells(rrbeg, 29) = Sheets(shsol).Cells(rrbeg, 17)
        Sheets(shsol).Cells(rrbeg, 30) = Sheets(shsol).Cells(rrbeg, 18)
        For j = 0 To ncomp - 1
          If xinitial.comp_cnt(i, j) > 0 Then
            Sheets(shsol).Cells(sumrowbeg + jpositive, 26) = i + 1
            Sheets(shsol).Cells(sumrowbeg + jpositive, 27) = j + 1
            Sheets(shsol).Cells(sumrowbeg + jpositive, 28) = xinitial.comp_cnt(i, j)
            Sheets(shsol).Cells(sumrowbeg + jpositive, 29) = mycomp(j).rel
            Sheets(shsol).Cells(sumrowbeg + jpositive, 30) = mycomp(j).cost
            jpositive = jpositive + 1
          End If
```

```
      Next j
   Next i
End Sub
```

Following module, named UtilityFunctions, is used for special functions used in previous modules.

```
Function INITIAL_SYSREL(ByVal k As Integer, ByVal n As Integer, ByVal compix As
Integer)
   INITIAL_SYSREL = 0
   If k = 1 Then
      INITIAL_SYSREL = (1 - (1 - mycomp(compix).rel) ^ n)
   Else
      For i = k To n
         INITIAL_SYSREL = INITIAL_SYSREL + COMBIN(n, i) *
(mycomp(compix).rel) ^ i * (1 - mycomp(compix).rel) ^ (n - i)
      Next i
   End If
End Function
Function COMBIN(ByVal n As Integer, ByVal k As Integer)
   COMBIN = FACT(n) / (FACT(k) * FACT(n - k))
End Function
Function FACT(ByVal i As Integer)
   FACT = 1
   If i = 0 Or i = 1 Then
      FACT = 1
   Else
      For j = 2 To i
         FACT = FACT * j
      Next j
   End If
End Function
Function ARRAY_SYSREL(ByRef arr_csr2() As Double)
   ARRAY_SYSREL = 1
   For i = 0 To nsubs - 1
      ARRAY_SYSREL = ARRAY_SYSREL * arr_csr2(i)
   Next i
End Function
Function ARRAY_SYSCOST(ByRef arr_csc2() As Double)
   ARRAY_SYSCOST = 0
   For i = 0 To nsubs - 1
      ARRAY_SYSCOST = ARRAY_SYSCOST + arr_csc2(i)
   Next i
End Function
Sub VerifyKofN()
```

```
    k = 2
    n = 8
    rel = 0.9
    myrel = 0
    For i = k To n
       myrel = myrel + COMBIN(n, i) * (rel) ^ i * (1 - rel) ^ (n - i)
    Next i
End Sub
```

Appendix B. 100 Pure Parallel Example Results

	GOZEBE HEURISTIC				MARGINAL ANALYSIS			
NO	RELIABILITY	COST	COMP. NO	INITIAL	RELIABILITY	COST	COMP. NO	INITIAL
1	0.99131	$29,486.1	43		0.99131	$29,486.1	43	
2	0.98476	$29,651.0	41		0.98145	$29,620.0	43	
3	0.98435	$29,980.4	41		0.98435	$29,980.4	41	
4	0.99091	$29,531.9	41		0.98984	$29,942.3	43	
5	0.98875	$29,613.5	42		0.98839	$29,541.0	42	
6	0.99777	$29,997.8	40		0.99777	$29,997.8	40	
7	0.98973	$29,868.5	42		0.98974	$29,790.8	42	
8	0.94776	$29,854.9	43		0.94776	$29,854.9	43	
9	0.99648	$29,718.6	38		0.99652	$29,755.9	38	
10	0.99042	$29,656.9	45		0.99042	$29,656.9	45	
11	0.98348	$29,853.0	42		0.98348	$29,853.0	42	
12	0.98139	$29,988.0	43		0.98139	$29,988.0	43	
13	0.98074	$29,646.3	41		0.98053	$29,562.2	41	
14	0.99507	$29,886.1	39		0.99507	$29,886.1	39	
15	0.99006	$29,579.1	41		0.99006	$29,579.1	41	
16	0.99468	$29,842.0	39		0.99468	$29,842.0	39	
17	0.99712	$29,685.0	41		0.99712	$29,685.0	41	
18	0.98618	$29,685.3	40		0.98618	$29,685.3	40	
19	0.98673	$29,999.8	42		0.98699	$29,896.3	42	
20	0.99495	$29,717.3	42		0.99509	$29,670.2	42	
21	0.98643	$29,831.0	41		0.98556	$29,941.0	41	
22	0.98551	$29,726.4	40		0.98223	$29,892.1	40	
23	0.98457	$29,896.4	42		0.98457	$29,896.4	42	
24	0.99311	$29,627.0	41		0.99296	$29,530.3	41	
25	0.99561	$29,895.8	42		0.99561	$29,895.8	42	
26	0.9682	$29,828.3	42		0.9682	$29,828.3	42	
27	0.99453	$29,781.7	44		0.99453	$29,781.7	44	
28	0.98659	$29,625.4	41		0.9857	$29,664.1	41	
29	0.9929	$29,584.1	43		0.99287	$29,511.8	43	
30	0.99067	$29,882.3	40		0.99067	$29,882.3	40	
31	0.99288	$29,945.5	43		0.99262	$29,788.4	43	
32	0.99583	$29,739.9	41		0.99502	$29,496.3	41	
33	0.993	$29,599.9	41		0.99307	$29,896.4	42	
34	0.99548	$29,629.5	43		0.99552	$29,635.1	43	
35	0.99309	$29,941.2	41		0.99329	$29,803.3	41	
36	0.9927	$29,533.3	38		0.99282	$29,558.2	38	
37	0.99797	$29,642.2	38		0.99797	$29,642.2	38	
38	0.98952	$29,971.3	44		0.98931	$29,874.6	44	
39	0.99531	$29,944.0	44		0.99531	$29,944.0	44	
40	0.99845	$29,653.7	40		0.99719	$29,519.0	42	
41	0.98613	$29,928.6	40		0.98294	$29,833.7	42	
42	0.98798	$29,912.6	38		0.98798	$29,912.6	38	
43	0.99061	$29,821.2	42		0.99061	$29,821.2	42	
44	0.99627	$29,969.3	39		0.99538	$29,807.9	40	
45	0.98988	$29,601.9	41		0.98988	$29,601.9	41	
46	0.99321	$29,985.1	42	**B.R.**	0.99297	$29,945.0	42	**B.R.**
47	0.99568	$29,693.8	41		0.99568	$29,693.8	41	
48	0.99464	$29,987.1	40		0.99381	$29,999.2	40	
49	0.99755	$29,866.6	36		0.99638	$29,999.6	39	
50	0.9933	$29,693.7	42		0.9933	$29,693.7	42	

51	0.99547	$29,919.0	43		0.99547	$29,919.0	43	
52	0.99632	$29,787.3	41		0.99468	$29,605.7	42	
53	0.99539	$29,807.2	44		0.99522	$29,752.4	44	
54	0.99961	$29,622.0	42		0.99962	$29,578.1	42	
55	0.99674	$29,701.6	42		0.99674	$29,701.6	42	
56	0.99501	$29,901.0	42		0.99501	$29,901.0	42	
57	0.98944	$29,953.5	45		0.98944	$29,953.5	45	
58	0.99232	$29,548.7	40		0.99232	$29,548.7	40	
59	0.99022	$29,940.5	42		0.99022	$29,940.5	42	
60	0.99014	$29,853.9	41		0.99045	$29,872.8	41	
61	0.98937	$29,552.7	37		0.98589	$29,573.5	38	
62	0.99407	$29,880.8	40		0.99416	$29,694.0	40	
63	0.9977	$29,907.9	41		0.99738	$29,847.3	42	
64	0.99537	$29,963.9	39		0.99521	$29,882.8	39	
65	0.98924	$29,909.8	42		0.98916	$29,913.2	42	
66	0.98432	$29,810.4	43		0.98128	$29,824.4	44	
67	0.99071	$29,951.6	40		0.98838	$29,682.6	41	
68	0.98655	$29,869.1	42		0.98655	$29,869.1	42	
69	0.97712	$29,818.6	45		0.97733	$29,697.7	45	
70	0.98109	$29,973.8	40		0.98109	$29,973.8	40	
71	0.98666	$29,504.4	43		0.98666	$29,504.4	43	
72	0.97662	$29,543.7	41		0.97825	$29,998.8	42	
73	0.99171	$29,602.2	41		0.99171	$29,602.2	41	
74	0.99319	$29,660.2	39		0.99328	$29,559.8	39	
75	0.99047	$29,882.8	41		0.99074	$29,866.1	41	
76	0.98982	$29,944.5	42		0.98982	$29,944.5	42	
77	0.99811	$29,671.1	41		0.99811	$29,671.1	41	
78	0.99611	$29,619.7	40		0.99611	$29,619.7	40	
79	0.99342	$29,792.3	43		0.99342	$29,792.3	43	
80	0.99791	$29,715.7	40		0.99791	$29,715.7	40	
81	0.99555	$29,807.1	44		0.99555	$29,807.1	44	
82	0.98682	$29,877.4	40		0.9849	$29,840.2	41	
83	0.96949	$29,565.1	43		0.96949	$29,565.1	43	
84	0.99298	$29,657.6	38		0.99298	$29,657.6	38	
85	0.99616	$29,525.8	41		0.99598	$29,926.8	43	
86	0.99721	$29,861.6	43		0.99715	$29,613.4	43	
87	0.99807	$29,725.6	43		0.99805	$29,594.5	43	
88	0.99358	$29,776.3	43		0.99358	$29,776.3	43	
89	0.99095	$29,798.6	40		0.99013	$29,548.6	40	
90	0.99581	$29,692.7	39		0.99579	$29,663.8	40	
91	0.99376	$29,555.7	44		0.99376	$29,555.7	44	
92	0.97339	$29,928.8	51	**B.R.**	0.97221	$29,951.6	51	**B.R.**
93	0.99541	$29,927.2	41		0.99549	$29,732.3	41	
94	0.99428	$29,575.7	39		0.99129	$29,599.7	41	
95	0.99613	$29,979.3	41		0.99499	$29,565.5	42	
96	0.98554	$29,960.5	38		0.98554	$29,960.5	38	
97	0.98899	$29,933.9	43		0.98899	$29,933.9	43	
98	0.99202	$29,815.8	42		0.99226	$29,825.7	42	
99	0.99075	$29,564.8	41		0.99142	$29,838.9	42	
100	0.99742	$29,874.6	41		0.99742	$29,874.6	41	

NOTE: The blank cells under INITIAL column mean that the initial system is populated with component, which have maximum reliability value.

Appendix C. 50 Example Results with one k-out-of-n System

	GOZEBE HEURISTIC				MARGINAL ANALYSIS			
NO	RELIABILITY	COST	COMP. NO	INITIAL	RELIABILITY	COST	COMP. NO	INITIAL
1	0.99493	$29,830.9	39		0.99493	$29,830.9	39	
2	0.98143	$29,814.4	40		0.98143	$29,814.4	40	
3	0.99148	$29,880.6	43		0.99148	$29,880.6	43	
4	0.98019	$29,723.9	42		0.98032	$29,703.6	42	
5	0.98317	$29,882.5	43		0.98281	$29,720.2	43	
6	0.99405	$29,989.2	40		0.99084	$29,575.1	41	
7	0.99325	$29,700.2	39		0.99325	$29,700.2	39	
8	0.99892	$29,676.0	43		0.99892	$29,676.0	43	
9	0.99044	$29,917.3	43		0.99047	$29,897.9	43	
10	0.98188	$29,558.7	46	B.R.	0.98188	$29,558.7	46	B.R.
11	0.98378	$29,884.7	42		0.98044	$29,956.1	43	
12	0.99842	$29,939.4	38		0.99842	$29,939.4	38	
13	0.98993	$29,706.9	41		0.99015	$29,471.8	41	
14	0.97523	$29,996.5	42		0.97644	$29,640.3	49	B.R.
15	0.98859	$29,868.0	41		0.98859	$29,868.0	41	
16	0.97216	$29,749.5	43		0.97216	$29,749.5	43	
17	0.989	$29,758.6	38		0.989	$29,758.6	38	
18	0.99767	$29,989.9	42		0.99767	$29,989.9	42	
19	0.97578	$29,893.1	42		0.97578	$29,893.1	42	
20	0.99444	$29,924.7	44		0.99444	$29,924.7	44	
21	0.99394	$29,982.8	37		0.99167	$29,992.7	39	
22	0.96319	$29,991.5	36		0.95909	$29,702.7	36	
23	0.993	$29,547.4	36		0.99324	$29,611.5	36	
24	0.98401	$29,663.9	42		0.98442	$29,821.4	43	
25	0.98985	$29,655.5	42		0.9889	$29,525.8	42	
26	0.99267	$29,970.0	39		0.99016	$29,563.8	40	
27	0.98357	$29,591.9	37		0.98226	$29,559.9	37	
28	0.99733	$29,717.6	42		0.99733	$29,717.6	42	
29	0.99488	$29,873.8	39		0.99488	$29,873.8	39	
30	0.99797	$29,791.5	40		0.99797	$29,791.5	40	
31	0.9748	$29,716.1	42		0.9748	$29,716.1	42	
32	0.97336	$29,719.4	41		0.97336	$29,719.4	41	
33	0.99424	$29,929.4	41		0.99424	$29,929.4	41	
34	0.99297	$29,691.2	45	B.R.	0.99297	$29,691.2	45	B.R.
35	0.98928	$29,964.4	38		0.98928	$29,964.4	38	
36	0.98547	$29,565.1	39		0.98547	$29,565.1	39	
37	0.97079	$29,737.3	41		0.97226	$29,940.7	42	
38	0.98807	$29,517.0	41		0.98807	$29,517.0	41	
39	0.9671	$29,997.9	39		0.96129	$29,600.6	39	
40	0.9888	$29,769.0	42		0.98899	$29,991.9	43	
41	0.97433	$29,950.8	41		0.97464	$29,985.0	41	
42	0.9852	$29,621.7	39		0.986	$29,854.7	40	
43	0.99865	$29,529.2	39		0.99865	$29,529.2	39	
44	0.99064	$29,986.4	43		0.99064	$29,986.4	43	
45	0.98943	$29,967.0	41		0.98921	$29,864.6	41	
46	0.99601	$29,828.3	40		0.99605	$29,852.2	40	
47	0.9972	$29,946.9	40		0.9972	$29,946.9	40	
48	0.98963	$29,965.7	41		0.98963	$29,965.7	41	
49	0.97844	$29,550.7	39		0.97716	$29,992.5	41	
50	0.99396	$29,935.2	38		0.99349	$29,977.4	39	

NOTE: The blank cells under INITIAL column mean that the initial system is populated with component, which have maximum reliability value.

Appendix D. 50 Example Results with two k-out-of-n Systems

	GOZEBE HEURISTIC				MARGINAL ANALYSIS			
NO	RELIABILITY	COST	COMP. NO	INITIAL	RELIABILITY	COST	COMP. NO	INITIAL
1	0.98996	$29,842.5	39		0.98996	$29,842.5	39	
2	0.99323	$29,985.8	48	B.R.	0.99153	$29,745.4	49	B.R.
3	0.99743	$29,609.5	39		0.99761	$29,757.5	40	
4	0.98922	$29,837.8	39		0.98965	$29,732.2	39	
5	0.98706	$29,516.0	39		0.9887	$29,936.0	40	
6	0.98289	$29,966.3	40		0.97935	$29,642.5	40	
7	0.97378	$29,928.6	39		0.97378	$29,928.6	39	
8	0.97851	$29,523.2	48		0.97975	$29,955.1	49	
9	0.9869	$29,602.4	40		0.9864	$29,800.0	41	
10	0.95351	$29,584.5	39		0.9553	$29,787.2	40	
11	0.998464	$29,779.9	40		0.998463	$29,912.9	41	
12	0.95039	$29,543.1	47	B.R.	0.95039	$29,543.1	47	B.R.
13	0.9916	$29,535.2	36		0.99291	$29,874.6	37	
14	0.97542	$29,904.9	48		0.97502	$29,809.5	48	
15	0.99123	$29,935.9	41		0.99123	$29,935.9	41	
16	0.94788	$29,765.3	38		0.94788	$29,765.3	38	
17	0.99103	$29,766.3	38		0.99103	$29,766.3	38	
18	0.99241	$29,963.6	35		0.99213	$29,544.1	35	
19	0.96637	$29,647.9	36		0.96637	$29,647.9	36	
20	0.98785	$29,804.2	39		0.98783	$29,788.8	39	
21	0.97028	$29,813.5	39		0.97028	$29,813.5	39	
22	0.99323	$29,967.2	38		0.99323	$29,967.2	38	
23	0.9567	$29,808.5	41		0.95391	$29,554.6	41	
24	0.97995	$29,637.4	42		0.97711	$29,800.7	43	
25	0.98964	$29,775.9	41		0.98964	$29,775.9	41	
26	0.99265	$29,861.4	39		0.99265	$29,861.4	39	
27	0.98874	$29,823.9	37		0.98895	$29,838.8	37	
28	0.9913	$29,752.0	40		0.99145	$29,712.5	40	
29	0.95792	$29,458.7	37		0.95715	$29,969.0	38	
30	0.97474	$29,840.3	40		0.97474	$29,840.3	40	
31	0.99306	$29,925.3	43	B.R.	0.99306	$29,925.3	43	B.R.
32	0.98522	$29,494.1	39		0.98522	$29,494.1	39	
33	0.98746	$29,923.6	36		0.98675	$29,536.2	36	
34	0.9954	$29,976.0	38		0.995	$29,854.8	38	
35	0.99317	$29,587.6	39		0.99317	$29,587.6	39	
36	0.97646	$29,967.5	42		0.97646	$29,967.5	42	
37	0.98048	$29,717.3	39		0.98048	$29,717.3	39	
38	0.99188	$29,516.7	36		0.99188	$29,516.7	36	
39	0.99746	$29,757.4	38		0.99746	$29,757.4	38	
40	0.9962	$29,554.3	40		0.9962	$29,554.3	40	
41	0.99729	$29,928.4	38		0.99729	$29,928.4	38	
42	0.99868	$29,999.9	37		0.99868	$29,999.9	37	
43	0.9937	$29,929.7	45	B.R.	0.99396	$29,977.8	45	B.R.
44	0.95642	$29,603.7	40		0.95642	$29,603.7	40	
45	0.98904	$29,784.2	40		0.9891	$29,778.0	40	
46	0.98486	$29,903.3	35		0.98404	$29,612.8	35	
47	0.96234	$29,570.6	38	B.R.	0.96344	$29,575.2	34	
48	0.98365	$29,634.0	38		0.98365	$29,634.0	38	
49	0.98891	$29,795.0	38		0.98898	$29,701.1	38	
50	0.98153	$29,797.6	39		0.98153	$29,797.6	39	

NOTE: The blank cells under INITIAL column mean that the initial system is populated with component, which have maximum reliability value.

Appendix E. 50 Example Results with ten k-out-of-n Systems

	GOZEBE HEURISTIC				MARGINAL ANALYSIS			
NO	RELIABILITY	COST	COMP. NO	INITIAL	RELIABILITY	COST	COMP. NO	INITIAL
1	0.99393	$39,624.5	58	B.R.	0.99405	$39,598.7	58	B.R.
2	0.98347	$39,795.0	45		0.98347	$39,795.0	45	
3	0.98868	$39,964.1	42		0.98868	$39,702.8	42	
4	0.98267	$39,856.4	49	B.R.	0.98258	$39,600.7	49	B.R.
5	0.95632	$39,227.9	44		0.95632	$39,227.9	44	
6	0.99256	$39,269.9	44		0.99256	$39,269.9	44	
7	0.97808	$39,409.7	43		0.9779	$39,207.4	43	
8	0.98217	$39,569.3	57	B.R.	0.98217	$39,569.3	57	B.R.
9	0.99388	$39,444.2	44		0.99388	$39,444.2	44	
10	0.97498	$39,431.6	45		0.97618	$39,947.5	46	
11	0.98443	$39,957.2	44		0.98443	$39,957.2	44	
12	0.99093	$39,359.0	46		0.99093	$39,359.0	46	
13	0.98541	$39,512.8	57	B.R.	0.98568	$39,557.9	57	B.R.
14	0.95359	$39,338.4	46		0.95359	$39,338.4	46	
15	0.98718	$39,568.9	44		0.98697	$39,397.0	44	
16	0.99489	$39,977.7	44		0.99464	$39,733.9	44	
17	0.96541	$39,976.6	45		0.96541	$39,976.6	45	
18	0.9676	$39,996.6	44		0.9676	$39,996.6	44	
19	0.96994	$39,879.7	62	B.R.	0.96834	$39,607.5	62	B.R.
20	0.97985	$39,565.8	44		0.97985	$39,565.8	44	
21	0.96753	$39,962.3	59	B.R.	0.96753	$39,962.3	59	B.R.
22	0.9954	$39,926.0	45		0.9954	$39,926.0	45	
23	0.91549	$39,708.2	61	B.R.	0.91823	$39,630.0	61	B.R.
24	0.97781	$39,561.2	43		0.97781	$39,561.2	43	
25	0.98396	$39,701.2	44		0.98396	$39,701.2	44	
26	0.98743	$39,201.3	43		0.98743	$39,201.3	43	
27	0.96218	$39,279.5	43		0.96287	$39,963.7	44	
28	0.98006	$39,435.0	43		0.98006	$39,435.0	43	
29	0.98056	$39,518.7	44		0.98056	$39,518.7	44	
30	0.99313	$39,780.6	45		0.99314	$39,687.0	45	
31	0.99417	$39,990.0	44		0.99484	$39,994.2	44	
32	0.99283	$39,504.3	44		0.99283	$39,504.3	44	
33	0.993	$39,490.7	45		0.993	$39,490.7	45	
34	0.97732	$39,859.0	45		0.97732	$39,859.0	45	
35	0.91194	$39,521.2	42		0.91558	$39,886.7	43	
36	0.98062	$39,398.4	43		0.98049	$39,302.4	43	
37	0.99179	$39,876.1	45		0.99179	$39,876.1	45	
38	0.96971	$39,546.6	54	B.R.	0.96971	$39,546.6	54	B.R.
39	0.9994	$39,732.2	45		0.9994	$39,732.2	45	
40	0.99789	$39,566.6	44		0.9979	$39,362.3	44	
41	0.98713	$39,967.2	45		0.98703	$39,838.3	45	
42	0.99479	$39,608.7	45		0.99479	$39,608.7	45	
43	0.99049	$39,547.9	43		0.9907	$39,387.9	43	
44	0.97391	$39,596.5	42		0.97391	$39,596.5	42	
45	0.99096	$39,604.5	57	B.R.	0.99096	$39,604.5	57	B.R.
46	0.97555	$39,979.5	43		0.97555	$39,979.5	43	
47	0.96233	$39,439.1	45		0.96233	$39,439.1	45	
48	0.97317	$39,600.9	46		0.97306	$39,906.8	47	
49	0.9475	$39,957.0	64	B.R.	0.94301	$39,509.6	64	B.R.
50	0.95957	$39,518.8	62	B.R.	0.95957	$39,518.8	62	B.R.

NOTE: The blank cells under INITIAL column mean that the initial system is populated with component, which have maximum reliability value.

Appendix F. 10 Example Results of LINGO Optimization Tool

LINGO SOLUTIONS OF THE FIRST TEN QUES.					
	RELIABILITY	COST	# of COMP	TIME	ITERATIONS
1	0.9971622	29962.97	39	7:35	218370
2	0.9935464	29955.32	35	1:14	34950
3	0.9924712	29970.59	40	9:53	273524
4	0.9967201	29981.14	37	2:22	67288
5	0.9952257	29988.98	38	4:12	124688
6	0.9990959	29991.68	34	5:32	149798
7	0.9976085	29987.84	43	0:32	16181
8	0.9662721	29978.41	41	30:43	529592
9	0.9984189	29914.63	36	0:52	19241
10	0.9960495	29984.23	38	1:51	36632

Bibliography

1. Reeves Colin R. "Modern Heuristic Techniques for Combinatorial Problems". UK: McGraw-Hill, 1995

2. Muller Heiner and Merbach "Heuristics and Their Design: A Survey", *European Journal of Operational Research*, 8: 1-23 (1981)

3. Dhillion Balbir S. "Quality Control, Reliability, and Engineering Design". New York: M.Dekker, 1985

4. Hoyland Arnljot and Rausand Marvin "System Reliability Theory Models and Statistical Methods". New York: John Wiley & Sons, 1994

5. Jones James V. "Engineering Design: Reliability, Maintainability, and Testability". PA: Tab Books, 1988

6. Ebeling Charles E. "An Introduction to Reliability and Maintainability Engineering". New York: McGraw-Hill, 1997

7. Elsayed Elsayed E. "Reliability Engineering". Massachusettes: Addison Wesley Longman, Inc., 1996

8. Dhingra Anoop K. "Optimal Apportionment of Reliability & Redundancy in Series Systems Under Multiple Objectives", *IEEE Transactions on Reliability*, Vol. 41, No. 4, (December 1992)

9. Ushakov Igor and Harrison Robert "Handbook of Reliability Engineering". New York: Wiley, 1994

10. Albright Christian S. "VBA for Modelers: Developing Decision Support Systems with Microsoft Excel". California: Duxbury, 2001

Vita

First Lieutenant Huseyin Gozebe attended Turkish Land Force Academy in August 1990. He graduated from Academy in August 1994 as a system engineer. He continued his education at Construction Management School of Turkish Air Force between 1994-1995. He finished top in his class. He was assigned to the Turkish Air Force Academy in 1995, and worked at the Academy as a construction manager between 1995-2001.

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 074-0188

1. REPORT DATE *(DD-MM-YYYY)*	2. REPORT TYPE	3. DATES COVERED *(From – To)*
14-03-2003	**Master's Thesis**	Sep 2002 - Mar 2003

4. TITLE AND SUBTITLE
OPTIMUM COMPONENT DESIGN IN N-STAGE SERIES SYSTEMS TO MAXIMIZE THE RELIABILITY UNDER BUDGET CONSTRAINT

5a. CONTRACT NUMBER

5b. GRANT NUMBER

5c. PROGRAM ELEMENT NUMBER

6. AUTHOR(S)
Huseyin GOZEBE, First Lieutenant, TUAF

5d. PROJECT NUMBER

5e. TASK NUMBER

5f. WORK UNIT NUMBER

7. PERFORMING ORGANIZATION NAMES(S) AND ADDRESS(S)
Air Force Institute of Technology

8. PERFORMING ORGANIZATION REPORT NUMBER
AFIT/GOR/ENS/03-08

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)

10. SPONSOR/MONITOR'S ACRONYM(S)

11. SPONSOR/MONITOR'S REPORT NUMBER(S)

12. DISTRIBUTION/AVAILABILITY STATEMENT
APPROVED FOR PUBLIC RELEASE; DISTRIBUTION UNLIMITED.

13. SUPPLEMENTARY NOTES

14. ABSTRACT
This research is intended to find a new heuristic for solving the reliability optimization of n-stage series system. The new heuristic will make the initial system design and use redundancy to improve system reliabilility level. The limited capacity of common optimization tools requires the new heuristic coded by VBA programming language. Moreover, the known realibility optimization heuristic, marginal analysis will be coded so that the difference between two heuristics can be seen and design engineers can use the best result in their applications.

15. SUBJECT TERMS
Reliability, Reliability Block Diagrams, Reliabilty Optimization Tasks, Reliability Allocation Methods, VBA

16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT	18. NUMBER OF PAGES	19a. NAME OF RESPONSIBLE PERSON
a. REPORT	b. ABSTRACT	c. THIS PAGE			Stephen P. Chambal, Capt, USAF (ENS)
U	U	U	UU	119	19b. TELEPHONE NUMBER *(Include area code)* (937) 255-6565, ext 4314; e-mail: Stephen.Chambal@afit.edu

Standard Form 298 (Rev. 8-98)
Prescribed by ANSI Std. Z39-18